NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

METHODS FOR INTELLIGENT MAPPING OF THE IPV6 ADDRESS SPACE

by

Blake W. LaFever

March 2015

Thesis Advisor:	Robert Beverly
Second Reader:	Geoffrey Xie

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	03-27-2015	Master's Thesis 09-30-2013 to 03-27-2015

4. TITLE AND SUBTITLE

METHODS FOR INTELLIGENT MAPPING OF THE IPV6 ADDRESS SPACE

5. FUNDING NUMBERS

N66001-2250-58231

6. AUTHOR(S)

Blake W. LaFever

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of Homeland Security
245 Murray Lane SW, Washington, DC 20528

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Due to the rapid growth of the Internet, the available pool of unique addresses in version four of the Internet Protocol (IPv4) is nearly depleted. As a result, the next generation protocol, IPv6, is now widely implemented and rapidly being adopted. This thesis examines new methods for active mapping of the IPv6 topology, i.e., router and link discovery. Better characterization of the IPv6 topology can provide the Department of Defense and other federal agencies the ability to defend networks and more effectively respond to cyber attacks. However, given that the IPv6 address space is roughly 79 billion billion billion times larger than the IPv4 space, mapping in IPv6 introduces new network measurement challenges. As a first step toward intelligent IPv6 topology discovery, this thesis takes lessons learned in efficient IPv4 mapping algorithms and attempts to apply them to IPv6. We develop several novel IPv6 mapping techniques and evaluate their probing time and topological coverage as compared to current state-of-the-art systems. Finally, we uncover previously unrecognized properties of several IPv6 deployments, infer network topologies, and characterize common IPv6 subnetting structure.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Internet Topology, Network Topology, IPv6, IPv6 Topology, Adaptive Probing, Efficient Topology Discovery	79
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

METHODS FOR INTELLIGENT MAPPING OF THE IPV6 ADDRESS SPACE

Blake W. LaFever
Lieutenant Commander, United States Navy
B.S., Georgia Institute of Technology, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN CYBER SYSTEMS AND OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
March 2015

Author: Blake W. LaFever

Approved by: Robert Beverly
Thesis Advisor

Geoffrey Xie
Second Reader

Cynthia Irvine
Chair, Cyber Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Due to the rapid growth of the Internet, the available pool of unique addresses in version four of the Internet Protocol (IPv4) is nearly depleted. As a result, the next generation protocol, IPv6, is now widely implemented and rapidly being adopted. This thesis examines new methods for active mapping of the IPv6 topology, i.e., router and link discovery. Better characterization of the IPv6 topology can provide the Department of Defense and other federal agencies the ability to defend networks and more effectively respond to cyber attacks. However, given that the IPv6 address space is roughly 79 billion billion billion times larger than the IPv4 space, mapping in IPv6 introduces new network measurement challenges. As a first step toward intelligent IPv6 topology discovery, this thesis takes lessons learned in efficient IPv4 mapping algorithms and attempts to apply them to IPv6. We develop several novel IPv6 mapping techniques and evaluate their probing time and topological coverage as compared to current state-of-the-art systems. Finally, we uncover previously unrecognized properties of several IPv6 deployments, infer network topologies, and characterize common IPv6 subnetting structure.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

AFRINIC Internet Numbers Registry for Africa

ARIN American Registry for Internet Numbers

Ark Archipelago

AS Autonomous System

BGP Border Gateway Protocol

CAIDA Center for Applied Internet Data Analysis

CIDR Classless Inter-Domain Routing

DNS Domain Name System

DOD Department of Defense

IANA Internet Assigned Numbers Authority

ICMP Internet Control Message Protocol

IETF Internet Engineering Task Force

IP Internet Protocol

IPS Ingress Point Spreading

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

ISC Interface Set Cover

ISP Internet Service Provider

LCP Least Common Prefix

NLSDE National Lab of Software Development Environment

NAT	Network Address Translation
NANOG	North American Network Operators' Group
NSS	National Security Strategy
NPS	Naval Postgraduate School
RFC	Request for Comments
RIR	Regional Internet Registry
RSI	Recursive Subnet Inference
RSI4	IPv4 RSI
RSI6	IPv6 RSI
SCP	Subnet Centric Probing
ToD	Topology on Demand
TCP	Transmission Control Protocol
VP	Vantage Point
VPS	Vantage Point Spreading
WHOIS	Who is

Acknowledgments

I would first and foremost like to acknowledge my classmates for constantly challenging me and keeping me on my toes. Their questioning attitude and constant drive enabled me to successfully complete this program.

I would like to express my heartfelt gratitude to my advisor, Dr. Rob Beverly, for his exceptional guidance throughout this process. So many times I would get stuck, and he always knew what direction I should take to move forward. Thank you for your encouragement and advice during the whole of this research.

I would also like to thank Dr. Geoffrey Xie for taking the time to provide a second look at this research. His efforts are greatly appreciated.

Lastly, I want to thank my loving husband, Jimmy, for his unwavering support and dedication throughout these last 18 months. I could not have gotten through it all without you!

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

In the late 1970s and early 1980s, the Internet was nothing more than a few inter-connected networks (hence Inter-net) [1]. Hosts (endpoints) communicated over these networks using an early version of Internet Protocol (IP), [1]. Subsequently, Internet Protocol version 4 (IPv4) was developed and utilized as the standard communication protocol for inter-networking. With 2^{32} possible IP addresses, the standard was thought to be able to handle a global web of interconnected networks [2]. Just over a decade later, with the adoption and rapid expansion of the Internet, the Internet Engineering Task Force (IETF) forecasted the exhaustion of the IPv4 address space within two to three decades. This prompted the design of a new standard, Internet Protocol version 6 (IPv6), which expands the IP space to 2^{128} unique addresses.

A key area of study among researchers is how the IP space is being subdivided and utilized. Such inferences are frequently termed topology mapping or network mapping. Topology is simply the combination of nodes (e.g., interfaces or routers) and edges (links between them) in a network. Given the large size of the address space in both IPv4 and IPv6, it is favorable to explore efficient and accurate techniques for performing topology mapping. Several prior IPv4-efficient techniques and results will be discussed in Section 2.2.1. The research conducted here will focus on methods designed to accomplish similar goals in IPv6.

This chapter will discuss the applicability of this research to the Department of Defense (DOD), goals and objectives, challenges encountered, and provide a summary of contributions to advance progress in this area.

1.1 DOD Applications

In recent years, significant emphasis has been placed on expanding the DOD's capabilities in the area of cyber systems and operations. The mention of cyber in top-level policy over the last several years has begun to expand, indicating a shift of focus to the understanding and defense of the cyber domain.

In his 2015 State of the Union address, President Barack Obama stated:

> No foreign nation, no hacker, should be able to shut down our networks, steal our trade secrets, or invade the privacy of American families, especially our kids. We are making sure our government integrates intelligence to combat cyber threats, just as we have done to combat terrorism. [3]

In IPv4, that intelligence included a fundamental understanding of how networks are structured in order to better attack or defend that structure as necessary. With time, knowledge of an adversary's network structure can be leveraged into creating an effective cyber capability. Defensively, knowing how networks are interconnected allows for better protection of critical assets. Given the increasing adoption rate and vast IP address space of IPv6, the challenge lies in finding a way to reliably and accurately discover the topology of a network.

In the 2010 National Security Strategy (NSS), there is a section devoted to the security of cyberspace [4]. The two-way nature of the cyber domain not only allows the DOD to pursue its goals but also allows adversaries a means to attack us. In this strategy, President Obama lays out the importance of investing in cooperation between government and the public and private sectors to combat the challenge of securing networks [4]. No one entity will be able to address all threats to friendly systems; only with a coordinated effort and cooperation among all interested parties can effective defenses be created against cyber threats.

The U.S. is not the only country shifting to IPv6. Large-scale shifts are being seen in many countries abroad. Peru, Norway, Germany, China, Japan, Belgium, and Malaysia all show significant increases in IPv6 adoption, while several match or exceed the adoption rate of the U.S. [5]. Coupled with the exponential increase in the number of IPv6 Border Gateway Protocol (BGP)-announced prefixes across several countries [6], it is imperative that we fully understand the fundamentals of IPv6.

The first step to secure networks on the IPv6 level is to understand how these networks are constructed and interconnected. The research presented herein attempts to address this challenge and provide methods to help understand it.

1.2 Goals and Objectives

There is a wealth of existing IPv4 research on active topology mapping, specifically regarding efficient methods for mapping the IPv4 topology. Given the diverse implementation nature of IPv4 and methods used to extend its useful life on the Internet, many different techniques have been utilized to return results that are accurate and time efficient. The broad goal of this research is to develop active methods, based around IPv6 *traceroute* probes, that are efficient and timely in mapping the IPv6 address space. More specific to this thesis is to identify one efficient IPv4 probing algorithm and observe its performance on mapping networks within the IPv6 address space.

The first objective is to identify one efficient IPv4 probing algorithm and modify its code to operate on IPv6. This new algorithm should infer the subnetting structure of an IPv6 network, while balancing accuracy with timeliness. In this preliminary case, emphasis on accuracy rather than timeliness is preferable. In previous IPv4 work, several primitives were put forth as possible avenues for efficient methods to map the topology [7]. While each of the primitives proposed and tested enjoy different levels of success, this thesis will focus on the Recursive Subnet Inference (RSI) primitive to see how breaking subnets into subsequently smaller subnets affects IPv6 topology discovery.

The second objective is to compare results from this newly created IPv6 RSI (RSI6) technique to the results of a standardized IPv6 topology probing method. In this case, the well-regarded standard is the Center for Applied Internet Data Analysis (CAIDA) Archipelago (Ark) system of topology mapping [8]. It is important to note that IPv4 RSI (RSI4) was previously compared to the same Ark probing method.

Using the comparison results from IPv4 as a guide to interpret IPv6 results, the third objective is the iteration of RSI6 to improve IPv6 topology discovery. As data is gathered, IPv6 specific differences require further modification of RSI6 to improve the performance of IPv6 topology discovery.

Results in Section 4.2 led to investigation of exactly how the IPv6 address space is being subdivided, resulting in a fourth objective. The intent of this objective is to infer common practices for IPv6 subnetting. Through exhaustive probing of constituent /48s contained in a /32 prefix, knowledge of common subnetting behavior will be used to modify RSI6.

1.3 Challenges

The very nature of the Internet is challenging, especially from a topology mapping point of view. As Baltra [9] puts it: "A very distinctive characteristic of the Internet is its distributed structure with no centralized administration." This in itself presents large-scale issues when trying to determine a static picture of an entity that is constantly changing. The sheer size of the IPv6 address space adds to this challenge. With an address space $2^{128-32} \approx 7.9x10^{28}$ times larger than IPv4, the IPv6 address space is nebulous, ensuring that any brute-force method that attempts to map its topology will take inordinate amounts of time and will be out of date once complete.

As the number of systems that support IPv6 grows, the dynamic nature of routing changes the perceived topology. With IPv4, this was less of a concern because topologies of interest were typically bounded by an Autonomous System (AS) and the subnetting inside was limited by the number of bits remaining in the IPv4 address. The resulting number of subnet combinations can then be practically mapped. In IPv6, the number of subnet combinations inside an AS is now substantially larger, leading to many dynamic routing combinations, of which RSI6 may or may not interpret as differing subnets.

As of January 2015, Google IPv6 statistics reports that IPv6 networks comprise just less than six percent of Internet traffic generated by Google users [5]. While this percentage is up nearly three percent from a year ago and comprises a rather large number of systems, IPv6 is still in its infancy in terms of adoption across all Internet-connected systems. Given the relatively small percentage of systems currently requiring an IPv6 address, there is a large amount of "white space" where no networks or hosts exist. As described in Section 2.2.1, RSI6 continuously subdivides networks into subsequently smaller networks and probes each smaller network in an attempt to infer subnetting. RSI6 terminates when no new information is returned. Due to the large amount of unused space mentioned previously, RSI6 is likely to infer no subnetting when, in fact, subnetting exists elsewhere in the prefix.

A final challenge of this research is the lack of both ground truth knowledge and standardization of subnet practices. As mentioned previously, IPv6 adoption is still relatively low. With no centralized administration of its growth, Internet Service Providers (ISPs)

are free to configure allocated networks as necessary to provide IPv6 addresses to their constituents. Additionally, ISPs do not freely advertise their methods for subnetting their networks, making it difficult to apply human intuition to the RSI6 algorithm. Despite not knowing ground truth, relative comparisons between algorithms can provide an insight into overall performance.

As adoption of IPv6 continues to grow, the level at which some of these challenges affect researching efficient methods is expected to decrease, while other challenges will most likely arise. Standardization and common practices may become more commonplace while the dynamic nature of routing and persistent lack of ground truth may exacerbate the issue of topology mapping inside an AS. In any case, the size of the IPv6 address space will continue to challenge any method of topology mapping and remains the largest reason to continue research in this area.

1.4 Summary of Contributions

In this thesis, we explore the possibility of developing methods to effectively map topology in the IPv6 address space. This research will provide the following contributions:

1. Development of a RSI6 algorithm
2. Development of methods to help determine ground truth topology
3. Discovery of trends and validation of published best practices
4. Validation of a subset of ground truth findings with service providers

1.5 Thesis Structure

Subsequent chapters in this thesis are organized as follows:

- Chapter 2 reviews the differences between IPv4 and IPv6 and provides a summary of efficient topology mapping techniques and their results.
- Chapter 3 discusses the methodology of this research including the conversion of RSI to IPv6, subsequent iterations of RSI6, and application of ground truth methods.
- Chapter 4 details the results of applying RSI to IPv6, compares RSI6 results to Ark probing results, and validates ground truth testing with real-world knowledge.

- Chapter 5 provides conclusions and limitations of this research and recommendations for future work in this area.

CHAPTER 2:
Background and Related Work

To discuss the finer points of topology mapping, a basic understanding of particular aspects of the Internet is necessary. This chapter provides a general overview of the two main versions of IP, IPv4 and IPv6, and reviews relevant prior mapping research.

2.1 Background

The IPv6 address space, being approximately $7.9x10^{28}$ times larger than that of IPv4, presents a significant challenge with respect to mapping uniquely routable IP addresses. A brute-force method, for instance, would probe a large number of addresses inside a given prefix, attempting to discern its structure. Even if a brute-force method could accurately discover the topology, it would take an inordinate amount of time to complete due to the sheer size of the address space. In contrast, a method that probes only a few addresses per prefix would complete in substantially less time. The tradeoff in utilizing this method is then between the granularity of probing and the obtained accuracy of the network's structure. As current networks move toward IPv6 adoption, the depth of networks (i.e., subnetting) continue to increase, leading to a greater need for accurate, time-efficient methods for topology mapping.

2.1.1 IPv4 Address Space

Request for Comments (RFC) 791 provides the initial specification for IP [10]. One of the key design aspects of IPv4 is that each packet contains a 32-bit source and 32-bit destination address that identifies hosts on the Internet. This allows 2^{32} possible combinations for any IP address. The Internet works on the principle that all routable addresses must be unique, otherwise Internet traffic would not reach the intended recipient. Considering that, in 2013, the population of the world was roughly 7.125 billion people [11], 32 bits yields enough IP addresses for about 60% of the world population. Even if each person in the world used only one IP address, there would not be enough addresses to go around. Furthermore, in 2012, given approximately 8.7 billion devices connected to the Internet [12], it is clear that there are more devices requiring an IP address than there are people on the planet. The

inescapable conclusion here is that a 32 bit IP address does not provide enough unique addresses to connect each of these devices to the Internet.

More specifically, the pool of unallocated IPv4 addresses managed by the Internet Assigned Numbers Authority (IANA) was exhausted in February 2011 [13]. IANA allocates blocks of IP addresses to each Regional Internet Registry (RIR), who then assign portions of those blocks to requesting organizations within their geographic region. According to [13], three of the five RIRs have exhausted their pool of /8s (read "slash" eight) to assign. The American Registry for Internet Numbers (ARIN) is projected to exhaust their supply in May 2015 while the exhaustion of the Internet Numbers Registry for Africa (AFRINIC) is projected for March 2019 [13].

Classless Inter-Domain Routing (CIDR)

The original specification for IPv4 divided IP addresses into classes, which allowed vastly differing numbers of host addresses in each class. Class A, B, and C IP addresses are broken into network and local address portions based on the specific combination of the three highest order bits [14]. Class A addresses have 24 bits to specify a local address, yielding $2^{24} = 16$ million unique IP addresses within its network. Similarly, class B and C addresses have $2^{14} = 16k$ and $2^8 = 256$ unique IP addresses, respectively, inside each of their networks. Given the rigid structure of the classes, an ISP who requires slightly more IP addresses than a class B can provide would be given a class A block, which increases the number of usable host addresses by several million. In this case, it is unlikely that the ISP would ever use all 16 million addresses; therefore, many of those addresses remain unallocated, reducing the number of utilized IP addresses in IPv4 and contributing to the exhaustion of the IPv4 address space.

To address this issue, a short-term solution called CIDR was introduced in RFC 4632 in 1993 [15]. Using variable network lengths, denoted by the "slash notation" shown in Figure 2.1, a block of host IP addresses can be tailored more toward the needs of an organization, rather than assigning an overly large contiguous block where a majority of addresses go unused, as in classful addressing. Notice that each of the previously defined class addresses are included in the new scheme (i.e., /8, /16, /24), while each number of bits between can also be implemented.

notation	addrs/block	# blocks	
n.n.n.n/32	1	4294967296	"host route"
n.n.n.x/31	2	2147483648	"p2p link"
n.n.n.x/30	4	1073741824	
n.n.n.x/29	8	536870912	
n.n.n.x/28	16	268435456	
n.n.n.x/27	32	134217728	
n.n.n.x/26	64	67108864	
n.n.n.x/25	128	33554432	
n.n.n.0/24	256	16777216	legacy "Class C"
n.n.x.0/23	512	8388608	
n.n.x.0/22	1024	4194304	
n.n.x.0/21	2048	2097152	
n.n.x.0/20	4096	1048576	
n.n.x.0/19	8192	524288	
n.n.x.0/18	16384	262144	
n.n.x.0/17	32768	131072	
n.n.0.0/16	65536	65536	legacy "Class B"
n.x.0.0/15	131072	32768	
n.x.0.0/14	262144	16384	
n.x.0.0/13	524288	8192	
n.x.0.0/12	1048576	4096	
n.x.0.0/11	2097152	2048	
n.x.0.0/10	4194304	1024	
n.x.0.0/9	8388608	512	
n.0.0.0/8	16777216	256	legacy "Class A"
x.0.0.0/7	33554432	128	
x.0.0.0/6	67108864	64	
x.0.0.0/5	134217728	32	
x.0.0.0/4	268435456	16	
x.0.0.0/3	536870912	8	
x.0.0.0/2	1073741824	4	
x.0.0.0/1	2147483648	2	
0.0.0.0/0	4294967296	1	"default route"

Figure 2.1: CIDR chart, from [15]

Subnets

Subnets, or sub networks, are networks inside a larger defined network. The implementation of CIDR brings the ability to further subdivide a block of IP addresses. For example, if a company was assigned a /16 ($2^{16} = 64k$) block of addresses, they could choose to use two extra bits to subdivide their /16 into four ($2^2 = 4$) /18 blocks. This gives them four networks each containing $16k$ IP addresses. Additionally, each of those /18 blocks could be further subdivided into subsequently smaller networks.

Network Address Translation

In addition to CIDR, another short-term solution to the IPv4 address exhaustion problem is through re-use of addresses [16]. To do this, a general mapping function between uniquely routable IP addresses and re-used addresses was developed in 1994 [16], called Network Address Translation (NAT). NAT operates on the principle that there is a partition between re-used addresses and globally unique addresses, and NAT translates between them. The

advantage of this setup, in the context of address exhaustion, is that IP addresses can be re-used in local networks, while NAT translates requests to and from the globally unique IP space. Thus, NAT, combined with CIDR, has been successful in delaying the exhaustion of IPv4 addresses on the Internet.

2.1.2 IPv6 Terminology

To begin, some standard IPv6 terminology needs to be defined. In the binary world, one byte is equal to eight bits while a nibble is half of a byte, or four bits. We examine the prevalence of subnetting across non-nibble boundaries in Section 3.2.4. IPv6 addresses are typically represented in human readable form as a sequence of eight groups of two-bytes, where each group is written in hexadecimal and separated by a colon.

Another important property of IPv6 addressing is shorthand notation. Because of the length of IPv6 addresses, the formatting permits omission of leading zeros and runs of zeros. For example, the address 2601:0041:0011:0111:0011:0001:1111:0001 can be shortened to 2601:41:11:111:11:1:1111:1. The leading zeros have been removed, but are understood to be there. Additionally, the largest consecutive run of zeros from the address as a whole can be removed. For example, the address 2601:1141:0000:0000:0000:0000:0000:0001 can be written in shorthand as 2601:1141::1. Note that 23 nibble zeros have been removed and replaced with a double colon (::).

A reference to the ::1 address refers to the first address inside a given prefix, similar to the X.X.X.1 address in IPv4. This method of shorthand becomes important in Section 3.2.3 as RSI6 divides prefixes into smaller prefixes, each with its own unique ::1 address.

2.1.3 IPv6 Differences

The last IPv4 allocation of /8 prefixes from IANA to the RIRs was allocated in early 2011 [17]. Each RIR is rapidly approaching or has already exhausted its supply of IPv4 addresses allocated to customers [13]. IPv6 was created primarily to resolve the IPv4 address exhaustion problem through an increase in bits used for addressing [17, 18]. NAT and CIDR provided short-term solutions to prolong the usefulness of IPv4, however, the logical progression to a different standard is necessary as the number of networked systems continues to grow. Specifically, IPv6 has 128-bit addresses, expressed in human readable notation, as eight groups of four hexadecimal values separated by a colon (e.g.,

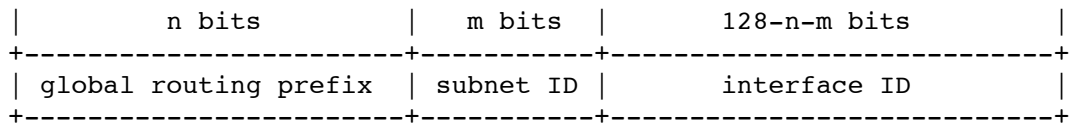


Figure 2.2: IPv6 address structure, from [19]

As with CIDR in IPv4, any number of bits (n+m) in an IPv6 address can be used as the network portion with the remaining 128-(n+m) being used as the unique interface identification. The separation between the two is denoted using the same "slash" notation introduced in IPv4. Common network blocks of IPv6 addresses are /32 or /48. Anything greater than /64 is generally reserved for end user management. By comparison, a single /64 network is $2^{64-32} \approx$ four billion times larger than the entire IPv4 address space, with enough IPv6 addresses for every person on earth to have their own /64 network.

2.1.4 IPv6 Adoption and Deployment

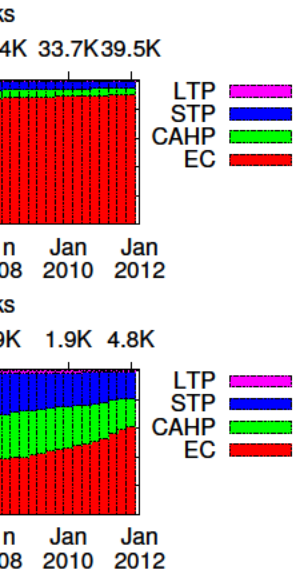
The IPv6 addressing standard, created in late 1995, was initially codified in RFC 1884 [20]. As lessons were learned from the deployment of IPv4, the standard went through several revisions. Roughly twenty years and several short-term IPv4 solutions later, adoption and deployment of IPv6 is becoming a reality.

In [21], the authors examine several aspects of measuring IPv6 adoption and deployment from several perspectives. Applicable to this research is the aspect of addressing, which the authors break into two steps: address allocation and network advertisement [21]. Before native IPv6 can be utilized by the Internet at large, IPv6 addresses must be allocated to all Internet backbone and downstream ISPs. The next step is to advertise those allocated prefixes, to ensure that users can reach a particular IPv6 destination. Additional aspects (e.g., naming, usage, and end-to-end reachability) are covered by [21] and contribute to the large-scale adoption and deployment of IPv6.

In [22], research shows an exponential growth rate in both AS nodes and links since 2007, where previous growth was slow and linear. The increased growth rate can be attributed to the implementation of native IPv6 by one of the largest backbone IPv6 providers, Hurricane

Electric [22]. Given the push toward IPv6, it is clear that adoption rates among downstream ISPs will continue to grow.

It is interesting to note that while growth is exponential over all IPv6 deployment, the rate of growth varies based on geographic location [21, 22]. In Figure 2.3, three of the RIRs growth rates for IPv4 and IPv6 are shown over a fourteen-year period. The x-axis provides a timeline in two- year increments, while the dual y-axes indicate the number of Autonomous Systems (ASs), in both IPv4 (on the left) and IPv6 (on the right). The dotted lines highlight IPv6 AS growth for three of the five RIRs during the specified timeframe. While all three are exponential post-2007, the rate of increase is significantly different. As IPv6 deployment and implementation continues to grow, it is expected that growth rates will continue to vary by geographic location.



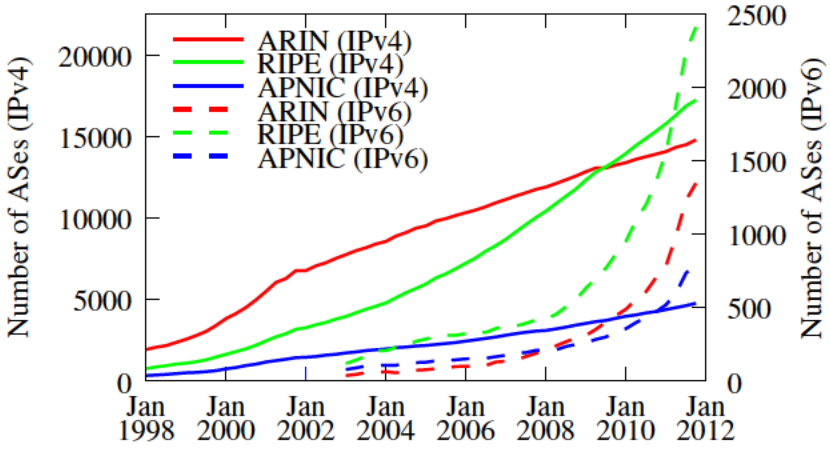


Figure 2.3: Growth by geographical region, from [22].

Figure 3: Growth in IPv4 and IPv6 ASes. RIPE overtook ARIN in the IPv4 graph in 2009; RIPE has always been ahead in IPv6.

fraction of EC ASes
% of the IPv6 graph,
, with ECs currently

ployment has primarily
, driven by transit and

business types in IPv6,
AS type in the IPv4 and
Ps and CAHPs all grow
6 graph has evolved dif-
s, we find that an initial
n data archiving began)
ential growth until the
he data. The exponents

2.2 Related Work

A significant research area among cyber experts is mapping the Internet in an effort to understand how nodes are interconnected. To do this, the majority of research is being conducted using active mapping techniques. In [23] and [24], two different systems implementing distributed networks of nodes are discussed. Each has a decentralized management structure which utilizes active probing originating at each node with results later aggregated in a central location for analysis [23, 24].

3.2 Growth trends by geographical region

Figure 3 shows the number of ASes in different geographical regions over time, according to the RIR WHOIS mappings described in Section 2. We omit the two smallest registries (LACNIC and AfriNIC), which have so few ASes compared to the three large registries (ARIN, RIPE and APNIC) that they are barely visible in the graph. The graph shows that for IPv4, the growth rate of RIPE-registered ASes has exceeded that of ARIN-registered ASes for the last decade (though both ARIN and RIPE showed linear growth in this period), and as of 2009 the RIPE region has more ASes than the ARIN region, a big difference from the early days of IPv4. For the IPv6 graph, on the other hand, the growth trend for each of the ARIN, RIPE and APNIC registries shows two distinct periods since 2003 – an initial linear phase

In previous studies, intelligent IPv4 active mapping techniques have shown a significant improvement in both accuracy and run time. Given that IPv6 operates in a similar manner to IPv4, applying proven techniques from IPv4 to IPv6 could provide similar benefits over standardized methods. This section will outline some of the IPv6 topology measurement systems available, provide qualitative results from selected IPv4 techniques as compared with other methods, and discuss related IPv6 topology discovery research.

2.2.1 IPv4 Intelligent Mapping

Each of the techniques discussed here focus on some specific topology-mapping challenge to enhance the amount of topology seen, whilst attempting to minimize the addition of time. With varying levels of success, each has shown that it is possible to improve performance over a recognized standard. Primarily, this section will cover the Ark platform and discuss the Subnet Centric Probing (SCP), Vantage Point Spreading (VPS), Recursive Subnet Inference (RSI), Ingress Point Spreading (IPS), Doubletree, and efficient tomography techniques.

CAIDA Ark

The Ark system employs the same techniques for IPv4 that were described previously for IPv6. It is recognized as the standard by which most intelligent topology techniques are compared. While other techniques utilize intuitive measures to get the entire topology at once, Ark relies on the accumulation of data over time to improve its topology database. Unfortunately, distributing probes over time inherently compares and integrates freshly discovered information with stale, and possibly incorrect, information. This can result in topologically incorrect mappings of specific prefixes.

Additionally, using a brute-force method, even to map just one prefix would generate an excessive amount of active probe traffic, possibly resulting in blockage of the probing machine. Consequently, the total amount of time it takes to completely map a prefix remains the same using either method. Ark avoids causing excessive network traffic, thereby remaining the standard by which other platforms are compared.

SCP/VPS

The techniques discussed here attempt to tackle the topology mapping challenge from different aspects. As a result, each is able to improve performance with a corresponding reduc-

tion in load. As seen previously and posited by the authors, a complimentary combination of techniques often provides the most accurate results [7]. The SCP technique works on the basis that not every address within a prefix needs to be probed to get an accurate topology depiction. Instead, a dynamic "distance" between addresses is determined and the corresponding address away from the last address explored is probed [7]. The premise is that the prefix has some number of subnets and that each subnet will have a different path. This method provides good coverage of topology once inside the target prefix [7]. Specifically, in testing, SCP was able to capture upwards of 90% of network topology using less than 60% of the load when compared with Ark-style probing.

Conversely, VPS assumes there are multiple points of entry into an AS where each will provide a different view of the topology [7]. Distributing probes across multiple Vantage Points (VPs) yields reliability, redundancy, and increased accuracy. In testing, it was shown that implementation of VPS does not appreciably increase load while enjoying a six percent increase in topology discovery [7].

RSI/IPS

RSI is essentially a form of SCP, modified to overcome some of its observed shortfalls in an attempt to determine the subnet structure of a prefix [25]. Through continual subdivision of a network (i.e., a /16 into two /17s and two /18s from each /17, etc), RSI attempts to infer subnetting using new interfaces seen as it probes the smaller sub-networks. Where new interfaces are seen, it infers a subnet. Conversely, where no new interfaces are seen, probing is terminated on that specific sub-network.

In [25], it was shown that intelligent VP selection was important to reduce the probes required to gain the same knowledge of a network structure. IPS takes information from previous rounds of probing and provides a list of VPs that would maximize probing results within a prefix [25].

Clearly complementary, the combination of intelligent VP selection and implementation of an efficient network mapping algorithm (RSI) provides a large improvement over standardized methods. Specifically, two tests of a combined RSI+IPS algorithm were conducted resulting in "*more* topological information" using half as many probes in half the time as standard Ark probing of the same prefixes [25].

Doubletree

In [26], the authors research the redundancy of probing across links to get to a destination and attempt to minimize it. More precisely, they focus on two aspects of redundancy – intra- and inter-monitor – and provide specific load and performance results [26]. Intra-monitor redundancy refers to the links surrounding a single monitor, whereas inter-monitor redundancy is focused on links that are common to multiple monitors [26]. Using these focus points and an intelligent algorithm (*Doubletree*) to balance them, the authors are able to "reduce measurement load by approximately 76% while maintaining interface and link coverage above 90%" [26]. The underlying principles and techniques used in *Doubletree* could be adapted to IPv6, specifically to the challenge of inter-monitor redundancy. Reducing the measurement load with any efficient algorithm can free up probing resources for use elsewhere.

Efficient Network Tomography

In an alternate approach, utilizing characteristics that describe the relationship between two hosts, or network tomography, enable the development of an efficient algorithm to complement standard methods for topology discovery [27]. To enable the use of this relationship, the authors group hosts in a Depth-First Search order, that is, according to the similarity of their infrastructure [27]. These similarities (e.g., packet loss or round trip times) and their relationship to the underlying tree structure, coupled with multicast and unicast addressing, are the basis for developing this efficient algorithm [27]. As multicast and unicast addressing are both large parts of the operational aspect of IPv6, this technique provides an excellent basis for adaptation in IPv6 methods.

In conclusion, this work on efficient techniques in IPv4 provides a promising starting point for application to IPv6. The operational symmetry between IPv4 and IPv6 allows future implementation of proven IPv4 techniques to IPv6. Additionally, the wealth of knowledge gained through trial and error in IPv4 will contribute to the development of methods in IPv6 while hopefully sidestepping some of the challenges encountered. Building on a solid foundation of IPv4 background, there is much research to be conducted utilizing not only the proven techniques of IPv4, but also the challenges and knowledge gained.

2.2.2 IPv6 Topology Measurement Systems

This section provides background information about IPv6 topology measurement systems and includes relevant results as applicable.

National Lab of Software Development Environment (NLSDE)

The NLSDE at Beihang University in Beijing, China, designed a system called Dolphin, which takes inputs from BGP monitors and databases, 6bone updates, Domain Name System (DNS) queries, and information from Google to build its database of prefixes to probe [28]. Using this information, a proprietary algorithm is used to create an IPv6 destination address list [28]. Given the size of the IPv6 address space, a subset of all addresses is selected. The Dolphin system takes this address list, updated daily based on information from previously mentioned external sources, and utilizes distributed "agents" (similar to the monitors used by the CAIDA in Section 3.1) to probe those addresses and receive information regarding the topology.

In 2005, IPv6 topology measurements discovered $\approx 11,600$ links in over $5,000$ IPv6 addresses, roughly one and a half times more than CAIDA's system [28]. Using additional information to intelligently select destination IPv6 addresses provides insight into future efficient topology mapping techniques. No subsequent information regarding the Dolphin project has been released and is assumed to have been abandoned.

Lumeta

Lumeta is a company that provides product suites to manage information technology environments. One product, called IPsonar, provides network topology using multi-protocol discovery, layer 2 and 3 topologies, and device fingerprinting. According to [29], the software looks at common network protocols, port responses, network packets, and uses network path tracing to provide the network administrators "on-demand, point-in-time, network situational awareness." Using "recursive network indexing," IPsonar is able to dynamically add new network infrastructure (i.e., routers, links, etc.) when it senses a change from the established picture [29]. Due to the proprietary nature of software, the exact methodology behind this product is unknown; however, given the scope and general techniques described, it is unlikely that it would be used for large-scale IPv6 probing.

Raytheon and BBN Technologies

Taking BGP prefixes and adding specific pieces of information in an attempt to discern topology is one method of improving topology discovery. BBN Technologies, in concert with Raytheon, is working on "scanning smarter" [30] in the IPv6 domain by adding information to BGP-announced prefixes. Specifically, they use information from random subnetting, Who is (WHOIS) registrations, and sequence completion in an attempt to discover new interfaces.

BBN started by adding four random bits to each BGP-announced prefix in an attempt to find subnetting [30]. The assumption is that some networks perform only small amounts of subnetting, and through random selection and a small increase in probing, additional topology would be discovered. This technique closely resembles methodology used in ground truth discovery discussed in Section 3.3. With this additional information, they saw a 5.6% gain in discovered interfaces per trace [30].

Next, BBN downloaded WHOIS information positing that additional information may be registered and provide more specific network data than is advertised in BGP [30]. Additional information gathered allowed researchers to build a list of likely subnets in a prefix and then use active probes to discover interfaces. Using this technique, an increase of 11% was seen over using BGP alone [30].

Sequence completion is based on sequential counting of all possible combinations in a base of numbers (i.e., zero to "F" in hexadecimal). With IPv6, addresses that differ by only a few bits can indicate subnetting. During probing, BBN took sources addresses from responding packets to infer addressing schemes from the target prefix [30]. Combining this sequence completion technique with BGP and WHOIS methodology, BBN was able to discover 26.9% more interfaces per trace than BGP alone [30].

Results from testing show that each method encompasses a different set of interfaces with only minor overlap yielding significant increases in both detail and depth compared with BGP prefixes alone [30]. Upon analysis of all interfaces discovered during testing, it was found that 26.6% of them appeared in BGP-based traces [30]. With the addition of WHOIS data and sequence completion techniques, the remaining 73.4% interfaces were found. One of the key observations made by the authors was that each technique uncovered different

network topology [30]. In order to accurately depict the network topology as a whole, combination of the techniques is required.

CAIDA Ark

The CAIDA Ark system performs continuous IPv6 *traceroute* measurements for "all announced IPv6 prefixes (/48 or shorter) once every 48 hours" [8]. The system has been gathering this data since late 2008 [8] and CAIDA makes this data publicly available. This system employs a globally distributed network of monitors to discover the topology of the Internet via Internet Control Message Protocol (ICMP)-based *traceroute* probes. The Ark system methodology is provided by the CAIDA website:

> Each Ark monitor probes all announced IPv6 prefixes (/48 or shorter) once every 48 hours. One probing pass through all announced prefixes is called a cycle. In each cycle, a monitor probes only a **single** random destination in **each** prefix. Different monitors probe prefixes in independently-chosen random orders and probe to an independently-chosen random destination in each prefix. Prefixes are randomly ordered in such a way that a given monitor never probes the same prefix within 16 hours across cycle boundaries (a monitor can never re-probe a prefix **within** the same cycle, by definition). [8]

Effectively, each Ark monitor chooses two host addresses inside a prefix to probe, namely a random address and the ::1 (first) address of the prefix, and records all interfaces along that path. Using a binary file format called "warts," all *traceroute* information is saved and made available for public download. The system compares new *traceroute* data to existing data to determine if a new interface is found, and if so, it is added to the topology.

CHAPTER 3:
Methodology

To accomplish the goal of devising an efficient method for mapping IPv6 topology, there must first be an infrastructure capable of performing active probing of the address space. First, to provide various entry points into prefixes of interest, a network of distributed nodes across the Internet is necessary. Second, each of these nodes must be able to perform basic IPv6 tasks such as *ping* and *traceroute* to probe addresses of interest inside a prefix. Lastly, the system must be able to record pertinent information gathered from probes for future analysis. In Section 3.1, such a system, capable of providing each of these, is described.

Development of an efficient algorithm, through combination of efficient techniques and a distributed probing system, goes through several revisions. RSI4 is first ported over to IPv6, dubbed RSI6, and run to ensure operability with IPv6 and gather baseline performance statistics. Using subsequent results, RSI6 was then modified and tested, with each iteration attempting to build on baseline data. After application of different techniques and subsequent revisions, the focus of the research shifted to gaining a a form of ground truth knowledge of IPv6 networks – attempting to work backwards and develop an algorithm once the underlying topology is better understood. Using exhaustive probing and validation from ISPs, a baseline ground truth can be generated and used to improve the efficient topology mapping algorithm.

Section 3.1 provides an overview of the Ark platform and the Topology on Demand (ToD) interface that is the focal point for the research conducted here. Section 3.2 provides an overview of several IPv4-proven techniques used as a basis for developing an efficient probing algorithm in IPv6. In Section 3.3, exhaustive probing methods to gain ground truth knowledge of a network are discussed.

3.1 CAIDA Archipelago (Ark) and ToD

The Center for Applied Internet Data Analysis (CAIDA) in San Diego, CA maintains a system of distributed IPv4 and IPv6 network measurement nodes throughout the world, called Ark, with the purpose of actively mapping the Internet's topology [31]. These nodes

(or monitors), shown in Table 3.1, can be individually utilized or collectively grouped to perform large-scale probing [31]. The Ark platform performs continual IPv4 and IPv6 *traceroute* measurements, as discussed in Chapter 2, and has the ability to send custom probes via the Topology on Demand (ToD) interface. Continual measurement results are stored in binary warts files and made available for public download, while ToD results are stored in formatted text files on the local user's machine.

ams-nl	ams2-nl	ams3-nl	ams5-nl	anc-us
bcn-es	bma-se	bre2-de	bwi-us	cbg-uk
cgk-id	cjj-kr	cph-dk	dub-ie	eug-us
hel-fi	her-gr	hkg-cn	iad-us	jfk-us
lax-us	mnl-ph	muc-de	nce-fr	oak-us
ory-fr	pao-us	per-au	san-us	sin-sg
sin2-sg	sjc2-us	sof-bg	sql-us	syd-au
tpe-tw	vie-at	yow-ca	zrh2-ch	

Table 3.1: CAIDA Ark IPv6-capable monitors, from [32]

The ToD interface, which is an application programming interface called tod-client, allows a user to send probes and receive results from the monitors directly without having to log in to each monitor or develop distributed code. Tod-client, provided by CAIDA, is written in the Ruby programming language and is designed for bulk-probing by authorized users located anywhere around the world. In this manner, the administration portion of probing is centralized and controlled by the user. To discern between users, tod-client uses unique session-ids, ensuring that probe results are delivered to the correct user, while not interfering with probe requests from other users.

To interact with monitors, tod-client requires specific information on how to perform the requested probing. At a minimum, a probe identification number, the name of an IPv6 monitor, the type of probe to conduct (i.e., *ping* or *traceroute*), and the destination of the probe are required. Using this information, Ark will direct the requested monitor to perform a probe to the specified IPv6 destination and return the results. Once the monitor has completed the operation, it communicates back to Ark which then uses the session-id associated with the probe to route the results back to the requesting user's ToD interface. Specifically, results from probes include hop data, whether the destination replied or not, round trip times, probe halt reason, and whether all hops to the destination were found.

The RSI6 algorithm employs ToD dynamically by crafting probe requests based on the technique discussed in Section 3.2.2.

Use of ToD provides several benefits to the researcher. The basic interface is very simple and easy to use, crafting user input into complex and distributed commands to geographically diverse monitors. While it can be used directly via a command line interface, tod-client was meant to be called from an existing script – like RSI6 – which can dynamically and intelligently select what inputs to send to ToD. This allows researchers flexibility to implement new techniques, while maintaining the integrity of the overall probing system. An additional advantage of ToD is that probes and results are sent and received asynchronously, allowing bulk-probing to take place while results are received and processed.

Consequently, application of probing through tod-client also has a few limitations. First, while tod-client is asynchronous, it tracks probes to a client via a mandatory 'session-id,' meaning that only one instance of the efficient algorithm can run at any one time. The limitation can be bypassed using multiple copies of the algorithm with different identifications, however, it is not user friendly in this respect. Another limitation of tod-client is that results from a previous probe batch can interfere with a newly created probe batch. Again, the 'session-id' is the culprit, making Ark think that the current tod-client session is waiting for these delayed results. In some cases, delayed results can interrupt the script and cause tod-client to fail, losing all results gathered since it started. Last, with a distributed network of nodes, it is likely that monitors will be unresponsive at times. Currently, there is no way for users to track when a monitor is down and remove it from the list of usable monitors. Probes sent to an unresponsive monitor remain in the queue and do not return to the tasking tod-client session. This can lead to delays and ultimately a failure in the tod-client interface, causing a loss of results.

While Ark has some serious limitations, there are no real alternatives that provide IPv6 probing and the flexibility of general programmability – two key features necessary to perform this research. Systems such as PlanetLab currently do not support IPv6, while others like RIPE Atlas are limited in measurement inputs and do not provide the flexibility needed to conduct our research [33, 34].

3.2 Efficient Probing Techniques

Using the Ark platform described in Section 3.1, different techniques will be examined and tested to determine their effects on probing as compared with standardized methods. This section covers principles evaluated and how they are applied in IPv6.

3.2.1 Least Common Prefix (LCP)

In [7] and [9], LCP is introduced as the mechanism by which addresses are chosen in SCP, the precursor to RSI. The principle makes the supposition that, after dividing a prefix in two, addresses in different halves of a prefix could possibly be in different subnets if *traceroute* reveals different paths. This is illustrated in Figure 3.1.

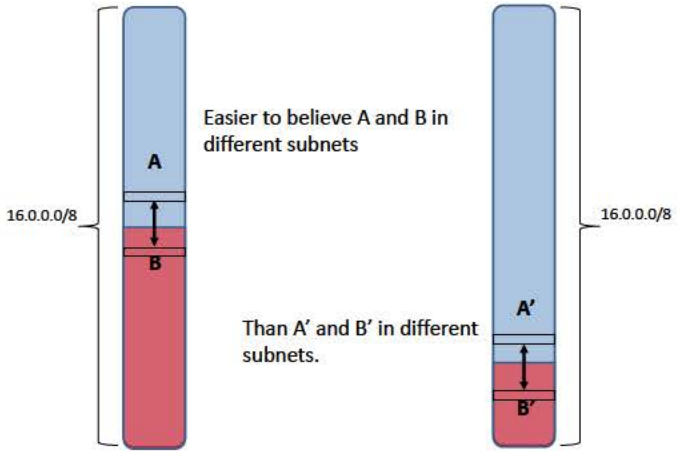


Figure 3.1: LCP in IPv4, from [9]

Further subdivision of a prefix yields a similar logic and is the basis of RSI. Specifically, RSI implements this concept continually as long as resulting probes return different information. Previous work shows that the LCP principle, applied in RSI, is very effective in actively mapping IPv4 network topology. In Section 3.2.3, the principle of LCP in IPv4 is applied to RSI in IPv6 with results presented in Chapter 4.

3.2.2 IPv6 RSI

The initial modification of the RSI4 algorithm to operate in IPv6 is relatively straightforward and requires only two major updates. First, where RSI4 references the 32-bit

nature of IPv4 IP addresses, RSI6 will reference 128 bits. Given that IPv6 operates in much the same way as IPv4 (e.g., set number of bits, variable "slash" notation, etc.), a few substitutions are all that is required to enable the code to interact with IPv6 networks.

Second, RSI6 must be directed to use CAIDAs IPv6-capable monitors for probing. Several of the monitors are dual use (IPv4/IPv6) while others are specifically one or the other. Table 3.1 lists CAIDA IPv6 monitors, current as of February 2015, but constantly changing as monitors are added. For best results, an up to date list of IPv6 monitors is kept within the algorithm. Every attempt is made to utilize all IPv6 monitors, however, unforeseen circumstances can cause a monitor to become unresponsive. Since the algorithm dynamically assigns probes to monitors, a method to detect downed monitors is implemented in later revisions to ensure only good monitors are used. With those modifications in place, IPv4 RSI is successfully ported over to IPv6 and ready for use. Table 3.2 provides basic RSI6 pseudocode showing how prefixes are dynamically split when new interfaces are found and terminated when no additional topological information is discovered.

Input: p/AS: set of prefixes / corresponding AS numbers
Output: R: probe result data

Algorithm: $RSI6(p/ASN)$

```
1:  int_list = ∅ // set of interfaces found
2:  for each prefix in p/AS do
3:     (dst) = select_dst(prefix)
4:     for each addr in dst do
5:        mon = random(list_of_mons)
6:        probe(dst,mon)
7:     R = read_results_from_Ark()
8:     if hop not in int_list then
9:        int_list+ = hop
10:       child_prefix = split(prefix)
11:       (dst) = select_dst(child_prefix)
12:       go to 4
13:    else
14:       return int_list
```

Table 3.2: Simple RSI pseudocode, after [9]

3.2.3 Targeted Probing within Prefixes

Minimizing probing time is one of the primary goals of developing an efficient algorithm. Clearly, increasing the number of probes to detect topology translates to an increase in the amount of time it will take to completely probe a prefix. Logically, the LCP principle discussed in Section 3.2.1 provides insight into how a prefix and its subnets are structured. As with IPv4 RSI, the base RSI6 algorithm probes only two addresses per prefix in an attempt to discover new topology. Mathematically, these addresses are located $\frac{1}{4}$ and $\frac{3}{4}$ of the way through a particular prefix space, shown in Figure 3.2. Based on the intuition of LCP, this is done to ensure that if subnetting were present, differing path interface hops results would be returned and RSI6 would continue subdividing the prefix. If the results of the two probes match, it is logical to conclude that no further subnetting exists in the prefix.

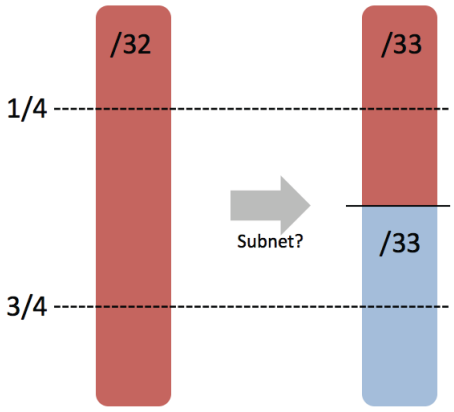


Figure 3.2: $\frac{1}{4}$ and $\frac{3}{4}$ target probes in IPv6

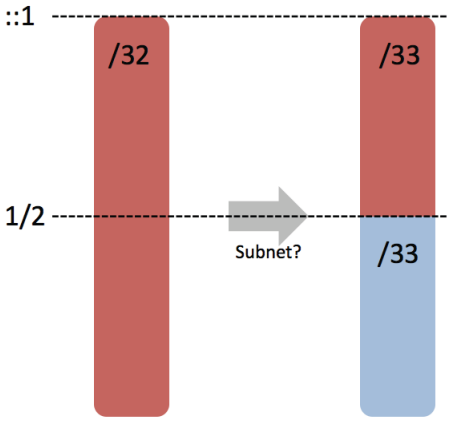


Figure 3.3: ::1 and $\frac{1}{2}$ target probes in IPv6

Given the supposition that an organization would begin issuing IPv6 addresses from the beginning of a prefix (i.e.,::1), some tweaking to the original RSI6 algorithm may provide additional topology with no added probing or time-cost. More specifically, probing the ::1 and the $\frac{1}{2}$ address provides results that test this supposition, while maintaining the ideology of LCP within a prefix. Figure 3.3 depicts an example prefix, broken into two halves, as specified by LCP, with the first IPv6 address of each probed. As before, any difference between results would indicate subnetting within the structure, whereas results with no difference would not.

In some cases, investing a little more time probing may result in larger gains. To test this theory, RSI6 is modified to combine the previous two targeted probing techniques, which may provide a large benefit with little cost. In this case, probes are targeted at the four points in the prefix, namely the ::1, $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ addresses, as shown in Figure 3.4. Should any of the four probes return differing results, RSI6 infers subnetting and continues to subdivide the prefix into evenly spaced constituents for further probing. It is important to note that probing in this manner will result in multiple probes to the same address. This happens with both the ::1 and $\frac{1}{2}$ addresses due to the prefix being split in two (corresponds to one bit). While this increases overhead, it is necessary to find subnetting in such a large IP space. Specific results for each targeted probing case are discussed in Section 4.1.1

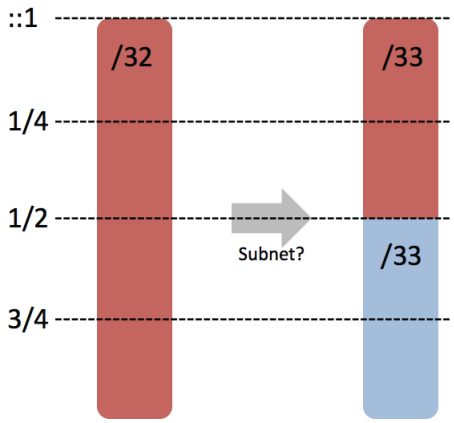


Figure 3.4: ::1, $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ target probes in IPv6

3.2.4 Subnet Boundaries and Best Practices

With 128 bits making up an IPv6 address, using base-10 numbers, as in IPv4, would make the corresponding representation of an IPv6 address quite difficult to work with. Additionally, base-10 numbers do not translate well when dealing with common practices of grouping bits, where groups usually consist of multiples of four, depending on the system. Using hexadecimal, bits in an IPv6 address are grouped in fours (a nibble) and then in groups of four nibbles separated by a colon (two bytes). The logical progression from bit to nibble to byte allows network administrators to conceptually subnet prefixes in IPv6 more easily. It follows that, a network administrator would subnet their assigned prefix in a location that makes logical sense, either on a nibble, a byte, or a two-byte boundary. Figure 3.5 gives two examples, the box on the left depicts two prefixes subnetted on a nibble

boundary. Notice the resulting subnets (/36s or /40s) count up incrementally corresponding to nibble separation. The box on the right depicts two prefixes which are not aligned to the nibble boundary, and thus, subnets (/37s or /41s), do not count incrementally. Depending on the needs and function of the organization splitting a prefix, the choice to split a prefix could be subnetting within the organization or the assignment of a group of subnets to customers. Subnets can then be further subnetted to support additional requirements.

Nibble Aligned Prefixes		Prefixes Not Nibble Aligned	
2001:db8::/32	**2001:db8::/32**	**2001:db8::/32**	**2001:db8::/32**
2001:db8::/36	2001:db8::/40	2001:db8::/37	2001:db8::/41
2001:db8:1000::/36	2001:db8:100::/40	2001:db8:800::/37	2001:db8:80::/41
2001:db8:2000::/36	2001:db8:200::/40	2001:db8:1000::/37	2001:db8:100::/41
2001:db8:3000::/36	2001:db8:300::/40	2001:db8:1800::/37	2001:db8:180::/41
<snip>	<snip>	<snip>	<snip>
2001:db8:c000::/36	2001:db8:9200::/40	2001:db8:6800::/37	2001:db8:8000::/41
2001:db8:d000::/36	2001:db8:9300::/40	2001:db8:7000::/37	2001:db8:8080::/41
2001:db8:e000::/36	2001:db8:9400::/40	2001:db8:7800::/37	2001:db8:8100::/41
2001:db8:f000::/36	2001:db8:9500::/40	2001:db8:8000::/37	2001:db8:8180::/41

Figure 3.5: Subnetting on nibble boundaries, from [35]

At a high level, the number of subnets is dictated by the requirement of the organization. Then, the appropriate number of bits is reserved as part of the CIDR mask for the network portion of the IPv6 address. With the large address space and infancy of IPv6, maximizing the utilization of bits used when subnetting is not yet important. Therefore, in [35], the North American Network Operators' Group (NANOG) recommends subnetting on the nibble boundary rather than per bit for human readability. For the smallest mask, the resulting subnets would contain plenty of host addresses and create $2^4 = 16$ subnets for the organization to work with. If more subnets are required, the organization can simply subnet on the next nibble boundary to increase the number of subnets and still have more than enough host addresses.

To test the theory that organizations are subnetting on nibble boundaries, RSI6 is modified to operate under this premise. The goal is to see whether this method can increase efficiency in detecting network topology without needing to probe non-nibble boundary addresses. In IPv4, RSI subdivides a prefix on the next least significant bit, effectively dividing the prefix into two equal halves. With IPv6 and nibble boundary subnetting, IPv6 RSI breaks a prefix

into its constituent 16 subnets (per nibble), which is more efficient than probing addresses on each bit boundary. The 16 subnets are grouped into fours (0-3, 4-7, 8-b, and c-f in hex) and one subnet from each is randomly chosen for probing. Figure 3.6 depicts this grouping.

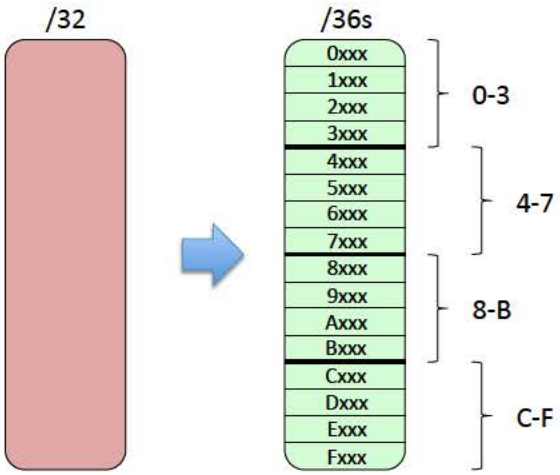


Figure 3.6: Subnet groups in RSI6

Not knowing where subnetting exists in a prefix, choosing a subnet to probe from each of the four equally-spaced groupings allows RSI6 the chance to find it regardless of location in the parent prefix. The intent is to account for nibble boundary subnetting without significantly increasing the number of probes necessary to detect topology across any part of the nibble. Specific results are discussed in Section 4.1.2.

3.3 Ground Truth

As research into efficient methods for probing IPv6 continues to develop, methods for determining whether discovered topology matches actual topology are necessary. The best method is to validate results with the provider managing the actual network structure. While this provides ground truth topology, the information can be difficult to obtain for a large number of prefixes due to the distributed administration and extensive allocation of the prefixes being researched. Lacking any ground truth knowledge of a prefix, a researcher is limited to relative comparisons between methods. This is a critical problem when attempting to evaluate the methods described in Section 3.2. This section discusses

obtaining a limited form of ground truth via exhaustive probing of /48s within /32s. Specifically, two methods of probing are attempted to achieve the ground truth:

1. ToD probing of top 20 prefixes as inferred via a simple bit-wise comparison program
2. Probing of $\approx$ 6,000 /32s directly using Ark monitors without ToD

3.3.1 ToD Probing

Developing a ground truth methodology requires scoping the problem down to a manageable subset of the IPv6 address space. This allows for faster testing of techniques and greater accuracy with which to gain some general insights about subnetting in IPv6. To do this, a single /32 prefix is broken down into $2^{16} = 65,536$ different /48s, with each /48 prefix's ::1 address being probed via ToD. The decision to partition prefixes in this manner was arbitrary, but made sense given that /32s and /48s are commonly assigned to customers. Additionally, focusing on the 16 bits between /32 and /48 provides a good sample base for testing without generating excessive probing traffic on target networks.

To select a single /32 for testing, eight months of historical Ark data from CAIDA and a simple inference program ranked each BGP-announced /32 by likely numbers of subnets. Table 3.3 provides pseudocode that determines numbers of subnets by examining destination probe addresses, grouping like prefixes, and then comparing hops along the path to the longest matching address. Interfaces with longer matching prefix lengths are inferred to be subnets.

The top 20 prefixes are then split up and probed; pertinent resulting information is stored in formatted text files for analysis. Each data file is then processed to extract interfaces seen inside the target AS during every individual *traceroute*. These interfaces are cataloged, counted, and graphed, with selected results discussed in Section 4.2.1.

3.3.2 Ark-direct Probing

To gain meaningful understanding of patterns across the Internet, data for all /32s is needed. Using ToD to perform ground truth analysis on 20 out of approximately 6,000 /32 prefixes provides insight into less than one percent of all /32s, which limits the scope of our insights. Due to its single-threaded nature and a limited timeframe, using ToD to perform probing on a large number of /32s is impractical.

Input: W: set of warts files
p/m: set of BGP-announced prefixes / masks
Output: R: set of subnets per prefix / mask

Algorithm: $infer\_subnets(W, p/m)$

```
1:  int_list = ∅ // set of interfaces found
2:  for each prefix in p/m do
3:    trie.node = prefix
4:  for each wartfile in W do
5:    for each trace in wartfile do
6:      for each hop in trace do
7:        if hop in target_AS then
8:          int_list = hop
9:      if int_list.length > 1 then
10:       found = find trace.dst in trie
11:       common_bits = compare int_list.hops to found.prefix
12:       if common_bits > found.prefixlength then
14:         found.subnets = +1
15: for each node in trie do
16:   R = node.prefix and node.subnets
17: return R
```

Table 3.3: Simple inference pseudocode, after [9]

Working with CAIDA, a method for exhaustively probing all /48s contained in every BGP-announced /32 directly from the distributed network of monitors was created. Using a script and a list of BGP-announced /32s, Ark randomly chooses roughly one hundred /32s, grouping them into a batch job. It then breaks each /32 up into constituent /48s, and randomly assigns one of 38 monitors to probe each /48, as depicted in Figure 3.7. Results from each *traceroute* are stored by the probing monitor in warts format. Once the batch of /32 prefixes is exhaustively probed, the individual monitors send one file containing their results to the central server, shown in Figure 3.8. The end result is a batch directory containing one warts file from each monitor, where the collective warts files hold complete results for approximately one hundred /32 prefixes. The data for all 6,000 /32s is contained in 51 directories (or batches).

A separate program to analyze the Ark warts files, shown in Table 3.4, is needed to parse

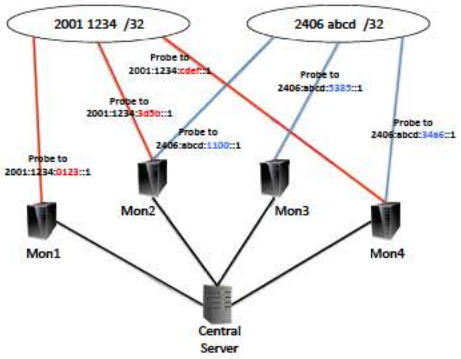


Figure 3.7: Representation of Ark bulk-probe methodology

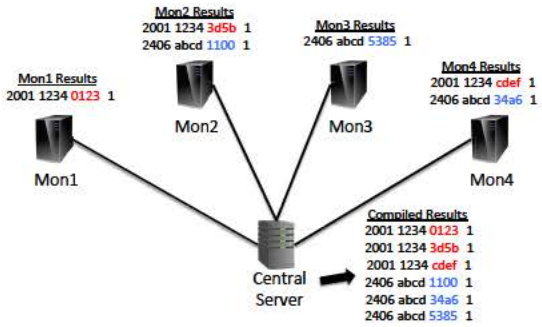


Figure 3.8: Representation of Ark monitor correlation of data

the raw data and extract useful metadata. The aforementioned metadata is stored in a Radix trie structure, where individual pieces of probe information can be analyzed. Specifically, interfaces seen along each *traceroute* path are compared with previously seen interfaces inside the AS. The first time an interface is seen, it is recorded, while subsequent occurrences are not. Counting unique interfaces contained in the trie provides a way to quantitatively infer subnets. In the ground truth method in Table 3.4, the program records a subnet if the cumulative interface count for a particular /48 jumps 1.5% over previous counts. The total number of subnets is then output upon completion of processing all /48s for a given /32.

An additional benefit to processing the interface data in this manner is that it can now be visually represented on a graph, either by interfaces seen by each /48 or a running tally of unique interfaces over all /48s. The different interfaces and their relative proximity lend some insight into possible subnetting in the prefix. Specific results and inferences from this methodology are discussed in Section 4.2.2.

Interface data gathered by Ark and subsequent processing of that data from all /32s on the Internet provides a basis to infer a limited form of ground truth. Using this ground truth, efficient probing techniques can be evaluated to ensure they are capturing the true subnetting present in IPv6 networks.

Input: W: set of warts files
p/m: set of BGP-announced prefixes / masks
Output: R: set of subnets per prefix / mask

Algorithm: $ark\_infer(W, p/m)$

```
1:  int_list = ∅ // set of interfaces found
2:  for each prefix in p/m do
3:    trie1.node = prefix
3:    trie2.node = prefix_48s
4:  for each wartfile in W do // begin ingesting traces
5:    for each trace in wartfile do
6:      for each hop in trace do
7:        if hop in target_AS then
8:          int_list = hop
9:      found_48 = find target_AS in trie2
10:     found_48.hops = int_list
11: for each slash_48 in trie2 do // begin counting/sorting ints
12:   count/sort 48_ints
13:   //begin inferring subnets
14:   if current_48_cnt > (prev_48_cnt + [1.5% of total 48_ints]) then
15:     parent_32 = find slash_48 in trie1
16:     parent_32.subnets = +1
17: for each node in trie1 do //begin output
17:   R = node.prefix and node.subnets
18: return R
```

Table 3.4: Ark probe analysis pseudocode, after [9]

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Results and Analysis

Specific results from testing both RSI6 and subsequent ground truth methods are described here, along with an analysis of findings. In Chapter 3, several methodologies were discussed attempting to improve the performance of the base RSI6 algorithm. While results from testing were largely inconclusive, several lessons learned will be carried forward into future work. Specific results of testing RSI6 are discussed in Section 4.1. Section 4.2 discusses results of ground truth exhaustive probing across single and collective prefixes, while identifying and characterizing patterns associated with the data to infer network structure. In Section 4.3, the results of ground truth testing for a select subset of prefixes are shared with managing organizations to determine the efficacy of our ground truth methods.

4.1 IPv6 RSI (RSI6)

The first iteration of RSI6 employed a simple one-for-one switch from IPv4 to IPv6 as described in Section 3.2.2. Subsequent iterations of RSI6 progressively incorporate techniques discussed in Section 3.2, with a consolidated view provided in Table 4.1. Note that monitor tracking was implemented for early versions of RSI6 in March 2015 testing due to unresponsive monitors.

	Base RSI6	*RSI6-v0*	*RSI6-v1*	*RSI6-v2*	*RSI6-v3*
$\frac{1}{4}$, $\frac{3}{4}$ Addresses	x		x	x	x
::1, $\frac{1}{2}$ Addresses		x			x
Nibble Bndry			x*	x	x
Monitor Trkng	**	**	**	x	x

** No pre-processing to align non-nibble prefix to nibble boundary*
*** Implemented for March 2015 testing*

Table 4.1: Iteration of RSI6 versus techniques implemented

To determine a baseline for relative comparison, this base RSI6 algorithm was given a list of approximately 19,000 unique BGP-announced prefixes to probe. Utilizing 18 IPv6-capable monitors, RSI6 randomly chose a monitor and issued *traceroute* probes via the tod-client interface.

Comparison between data gathered by Ark and RSI6 can be somewhat difficult. Ark data, for instance, is stored in individual warts files on a per-monitor and per-cycle basis. This equates to each warts file containing *traceroute* data for probes conducted to every BGP-announced prefix (roughly two probes per prefix). RSI6 results, by comparison, are dynamic and can encompass a varying number of probes to a single prefix. As such, every effort is made to provide results in a manner that presents an accurate apples-to-apples comparison.

Taking almost 22 hours and 170k traces to complete, the base RSI6 algorithm returned roughly 56k and 99k unique nodes and edges, respectively. By comparison, the Ark platform, performing the same number of probes over multiple monitors, discovered almost 42k unique nodes and just over 78k edges in a combined 220 monitor-hours. Normalization by the number of traces results in RSI6 discovering close to 32% more vertices and 24% more edges than Ark with a time reduction of over 90%. The consolidated results of each iteration are codified in Table 4.2 while a detailed comparison of RSI6 and Ark results is conducted in Section 4.1.3.

In an effort to verify the original results, the base version of RSI6 was re-run on the same 19,000 unique prefixes and results compared with those from Ark during a comparable timeframe. Approximately seven months later, the base RSI6 algorithm sent 143k probes and returned close to 48k and 76k unique nodes and edges in, which is significantly less than the original testing. This run of the base RSI6 algorithm also took roughly double the amount of time to completely probe all prefixes. Given the decrease in number of probes sent, the decrease in vertices and edges returned is expected. To confirm this, normalization of vertex and edge data with number of probes sent is done. Results are depicted in Figure 4.2 and are within a ten percent margin of error, signifying results are consistent with previous findings.

4.1.1 Targeted Probing

Recall from Section 3.2.3 that the base implementation of RSI6 probed target IPv6 addresses $\frac{1}{4}$ and $\frac{3}{4}$ into a prefix. As this was the case for RSI4, those fractions were ported directly over to RSI6.

Analyzing and comparing base RSI6 data against gathered measurement data from CAIDA

revealed that more interface data was discovered by probing the beginning address of a prefix. Consequently, this is one aspect of the methodology that Ark uses in its continual IPv6 probing. To test this hypothesis, the next iteration of RSI6, dubbed RSI6-v0, was created. In both the base and v0 versions of RSI6, the algorithm splits prefixes in half and probes one address in each half. Testing the hypothesis that there is more interface data at the beginning of a prefix, RSI6-v0 was modified to probe the ::1 and the $\frac{1}{2}$ address of a prefix, effectively targeting the starting address of each half of the parent prefix. For RSI6-v0, probes were generated and randomly assigned using the original 18 VPs.

Unfortunately, the resulting data from the first experiment was lost, however, running the same algorithm approximately seven months later resulted in $43k$ vertices and $69k$ edges seen in close to $129k$ traces. Finding over 96% of vertices and over 97% of edges compared with data collected by Ark during the same time period, RSI6-v0 accomplishes the same task as Ark with a 68% reduction in time to completion.

Logically, increasing the number of probes into a prefix should return more of that prefix's topology. As described in Section 3.2.3, combining the first two techniques by sending four probes per prefix should result in definite increases in both unique vertices and edges. Preliminary results, however, provided no substantial change from either of the previous approaches and this technique was not implemented in full-scale RSI6 probing until RSI6-v3 in favor of developing the nibble boundary technique.

4.1.2 Subnet Boundaries

Upon examination of previous results, the technique described in Section 3.2.4 was added to RSI6 in order to test the best common operating practice put forth by [35]. Using the base RSI6 probing method (i.e., probing addresses at $\frac{1}{4}$ and $\frac{3}{4}$), iteration RSI6-v1 was created to split a prefix on the nibble vice one bit, as in IPv4. As Table 4.1 notes, this technique is applied whether the announced prefix falls on a nibble boundary or not (i.e., a prefix mask divisible by four). For example, a /32 prefix will attempt to find subnetting in any of the constituent 16 /36s formed by subdividing the /32 with the next four bits. Both the /32 and constituent /36s fall on nibble boundaries. In contrast, a /33 prefix would be split into 16 /37s, neither of which fall on an easily readable boundary, making it less likely that a prefix would be subnetted there.

As noted in Section 3.2.4, subdividing a prefix in this manner to test the subnet boundary hypothesis introduced conflict between RSI6 termination conditions and generating excessive probing traffic. Specifically, in order to balance time constraints and the number of probes, not all constituent $2^4 = 16$ prefixes are probed with RSI6-v1. Instead, only a small fraction of the resulting 16 prefixes are probed. To select which prefixes get probed, recall from Figure 3.6 that each of the 16 prefixes are grouped in fours (e.g., 0x0-3, 0xC-F). One prefix from each is then randomly selected for probing, resulting in reducing the number of probes from 16 to four. If a new interface is discovered along any of the four probes, RSI6-v1 splits the returning prefix using the same methodology. This allows RSI6-v1 and subsequent iterations to uncover the tree-like structure in a given IPv6 prefix.

In the case of RSI6-v1, testing the nibble boundary hypothesis results in a $42k$ increase ($212k$ total traces) in probes sent compared with the base RSI6 algorithm. Randomly assigning traces to 18 monitors and implementing the nibble boundary technique, RSI6-v1 returns $40k$ and $74k$ unique vertices and edges, respectively – substantially less than seen with the base RSI6 algorithm. A second run of RSI6-v1, completed in March 2015, returned $38k$ and close to $63k$ vertices and edges, respectively, in over $203k$ probes. Results from this run of RSI6-v1 are consistent with the findings from 2014, with one notable exception. Edge count dropped roughly 11% from previous results, which could be attributed to more efficient routes developed on the Internet or the randomness associated with selecting Ark monitors for probing. Recall from Section 2.2.1 that, in IPv4, previous research proved that certain VPs will see more topology than others, resulting in differing results.

In an attempt to discern whether the decrease of returned topology between RSI6 versions was due to prefixes *not* aligned on a nibble boundary, pre-processing functionality of those prefixes was added to the next iteration of the algorithm, RSI6-v2. Specifically, any prefix with a prefix mask not divisible by four is split into multiple constituents of the next lowest mask that falls on a nibble. For example, a /33 is split into $2^{36-33} = 8$ constituent /36s where a /35 is split into $2^{36-35} = 2$ /36s. Each of these nibble-aligned prefixes is then added to the list of prefixes to be probed. RSI6-v2 conducts targeted probing of prefixes using the base RSI6 method (i.e., $\frac{1}{4}$ and $\frac{3}{4}$ addresses).

Implementing prefix pre-processing incurs a cost in number of prefixes to probe, specifically an increase of 45% (19,353 to 28,089 prefixes). As a result, the number of probes sent

by RSI6-v2 necessarily increases to match the added prefixes. With RSI6-v2, $44k$ vertices and nearly $91k$ edges were seen utilizing over $333k$ traces across 38 randomly selected VPs. The significant increase in VPs stems from an update to the most current list of monitors. The performance of RSI6-v2 is shown to deliver slightly better results than RSI6-v1, but noticeably less than the base RSI6 method. However, with a 95% increase in number of traces versus the base RSI6 algorithm with little improvement over RSI6-v1, additional modification is necessary.

Running RSI6-v2 in March 2015, using the same randomly selected monitors as before, resulted in $42k$ vertices and $77k$ edges in $308k$ traces. Again, results are consistent with the previous run of this RSI6 version, with a similar decrease in edges discovered, as in the March run of RSI6-v1. The same supposition holds true for this version of RSI6, as no intelligent VP selection technique is implemented.

An additional functionality added to RSI6-v2 and subsequent iterations is the tracking of downed or unresponsive monitors. Each RSI6 algorithm contains a listing of all IPv6-capable Ark monitors, to ensure an even distribution of load (to prevent overloading one monitor) and different vantage points for probing. When a probe is sent to a downed monitor, no notification is given to the user that the probe will not complete or return a result. Essentially, the probe request is queued waiting for the downed monitor to come back online. While the queuing function provides a necessary means to handle unresponsive monitors from an Ark platform perspective, it creates an inability for our algorithm to complete probing in a timely manner. In order to counter this, tracking is implemented in RSI6-v2 and later by keeping a running tally of probes sent to each monitor. When the program does not receive results for a specified time (roughly three minutes), it calculates the percentage of probes assigned to each monitor. If the number of outstanding probe requests to a monitor exceeds 10% of the total traces in flight, the monitor is temporarily removed from the list, and all probes sent to that monitor are randomly assigned to the remaining responsive monitors. Monitor tracking ensures that the time results of RSI6 testing are not artificially inflated by external delays.

In spite of the additional overhead cost associated with doubling the amount of probes per prefix, the final revision of the RSI6 algorithm, RSI6-v3, combines the targeted probing techniques of RSI6-v0 and RSI6-v1 with the nibble boundary pre-processing and monitor

tracking functionality of RSI6-v2. In effect, RSI6-v3 utilizes all efficient probing techniques discussed in Section 3.2 in one algorithm to probe prefixes across 38 randomly assigned VPs.

Initial probing in September 2014 resulted in 528k probes discovering 55k and 123k unique vertices and edges, respectively. RSI6-v3 performed considerably better than its previous revisions and produced similar vertices results and substantially more edge results compared to the base RSI6 algorithm. The increase in edges seen is most likely attributed to the 210% increase in probes sent over the base RSI6 method. While this is an improvement in results returned, an efficient algorithm is judged both by results and associated overhead costs – in this case, probe traffic.

Additional testing of RSI6-v3 approximately one and six months later were conducted with similar results. Specifically, in October 2014, RSI6-v3 saw 54k vertices and 121k edges in 604k probes sent across the same 38 randomly assigned VPs. Similarly, in March 2015, 52k and 103k vertices and edges, respectively, were discovered in 560k probes sent using 38 randomly assigned VPs. Vertices results are similar across all three test runs, whereas edge results, while similar for 2014 testing, reveal a significant drop during the 2015 run. It is unclear to what this decrease in edges can be attributed, however, future test runs may provide insight to whether the results from March 2015 were flawed in some way or represent a new norm.

Table 4.2 contains extensive and consolidated data regarding all RSI6 testing. Listed in two batches, each row provides the data for a particular run of RSI6 conducted. A total of five runs in two batches provide a starting point with which to compare this intelligent algorithm to a standard reference. Of note, the first batch, conducted between August and October 2014, lacks testing from RSI6-v0 while including a second run from RSI6-v3. Additionally, batch two, conducted in March 2015, contains one run from each iteration of RSI6, utilizing the same monitors and techniques discussed in previous sections. Analysis of the collective results with comparable Ark data is conducted in Section 4.1.3.

4.1.3 Analysis and Comparison of RSI6 Results

Before comparing RSI data against the well-regarded standard for IPv6 topology mapping, Ark, a relative comparison of how the standard has changed over six months is necessary.

	RSI6 Iteration	*# of VPs*	*Time to Complete*	*RSI6 Traces*	*RSI6 Vertices*	*RSI6 Edges*
2014	Base RSI6	18	21.5 hrs	170,522	56,463	99,184
	RSI6-v0	No Information Available				
	RSI6-v1	18	28.5 hrs	212,246	39,960	73,984
	RSI6-v2	38	45.5 hrs	333,346	44,249	90,874
	RSI6-v3 (1)	38	68.0 hrs	528,004	55,186	123,131
	RSI6-v3 (2)	38	82.0 hrs	604,108	53,944	121,100
2015	Base RSI6	18	42.5 hrs	143,242	48,364	76,524
	RSI6-v0	18	40.0 hrs	128,930	42,802	69,379
	RSI6-v1	18	49.0 hrs	203,126	38,049	62,874
	RSI6-v2	38	35.5 hrs	308,120	42,381	76,820
	RSI6-v3	38	47.0 hrs	559,740	52,336	102,824

Table 4.2: Consolidated RSI6 probing results

Table 4.3 provides data for five select monitors from September 2014 and March 2015.

	Monitor	*Cycle Time*	*Traces*	*Vertices*	*Edges*
Sept. 2014	cgk-id	44 hrs	33,358	26,799	31,258
	jfk-us	44 hrs	33,356	25,057	28,677
	lax-us	44 hrs	33,357	25,659	29,501
	sof-bg	44 hrs	33,351	25,250	30,552
	tpe-tw	44 hrs	33,356	26,787	31,164
March 2015	cgk-id	42 hrs	40,734	31,081	35,635
	jfk-us	42 hrs	40,732	29,051	32,854
	lax-us	42 hrs	40,727	29,412	33,374
	sof-bg	42 hrs	40,732	29,890	34,328
	tpe-tw	42 hrs	40,734	31,297	36,575

Table 4.3: Select Ark monitor data from September 2014 and March 2015

Since Ark provides results on a per-monitor and per-cycle basis, data from five separate monitors were chosen for comparison. The monitors were chosen based on geographically separated locations and availability of data from the necessary timeframe (in this case, early September 2014 and early March 2015). The cycle time refers to the amount of time it takes a monitor to completely probe the list of all BGP-announced IPv6 prefixes using the methodology discussed in Section 3.1. In 2014, the cycle time for each monitor was roughly 44 hours, whereas, in 2015, each took 42 hours to complete. There are a number of possibilities for the decrease including a reduction in number of prefixes, more efficient

probing routine, caching of results, and reduction in load due to an increased number of monitors. The cycle time of monitors contributes to the execution time comparison between Ark and RSI6.

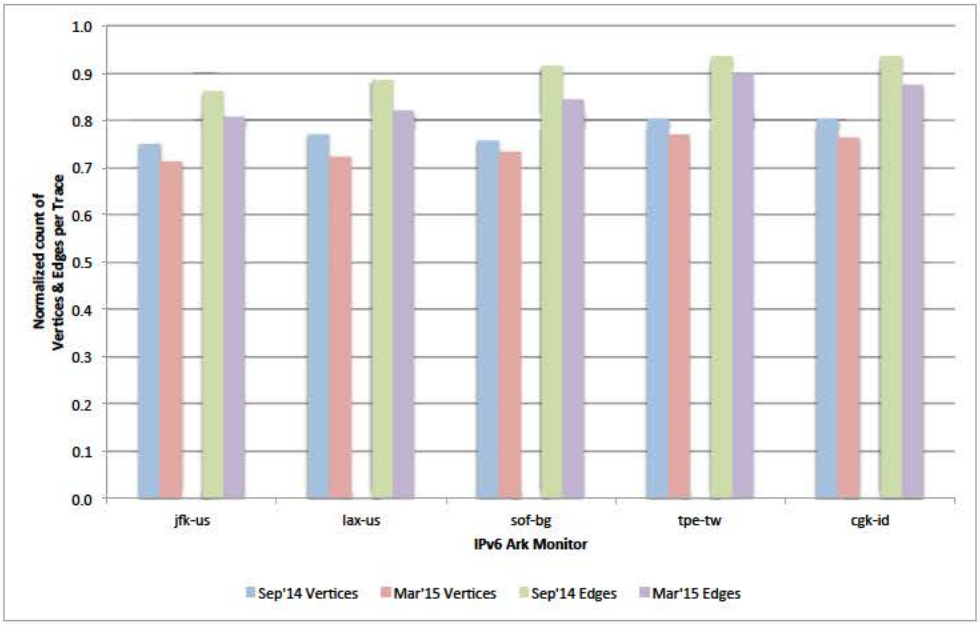


Figure 4.1: Normalized vertex and edge data generated by the Ark methodology for five select monitors

Comparing the number of traces with vertices and edges discovered, it appears that, in the six month time difference, the list of prefixes increased in size with a corresponding increase in vertices and edges. This increase makes logical sense due to the exponential increase in IPv6 adoption discussed in Section 2.1.4. An interesting point of discussion is that, when normalized per number of traces conducted, as in Figure 4.1, the data shows that there are fewer vertices and edges discovered than the prior round of probing. The most likely explanation seems to be an increase in number of announced prefixes with a lack of reachable destinations, probably due to increasing preparations for conversion to IPv6 in the wake of IPv4 exhaustion.

With a relative understanding of the evolution of the IPv6 landscape, an analysis between RSI6 and Ark can be conducted. Two specific metrics are used to analyze the performance of RSI6. First, a relative comparison of vertices and edges discovered by each method will reveal the ability of each probing method to discover the topology of IPv6 networks. In

Figure 4.2, the data presented in Table 4.2 is graphically displayed next to corresponding CAIDA Ark data.

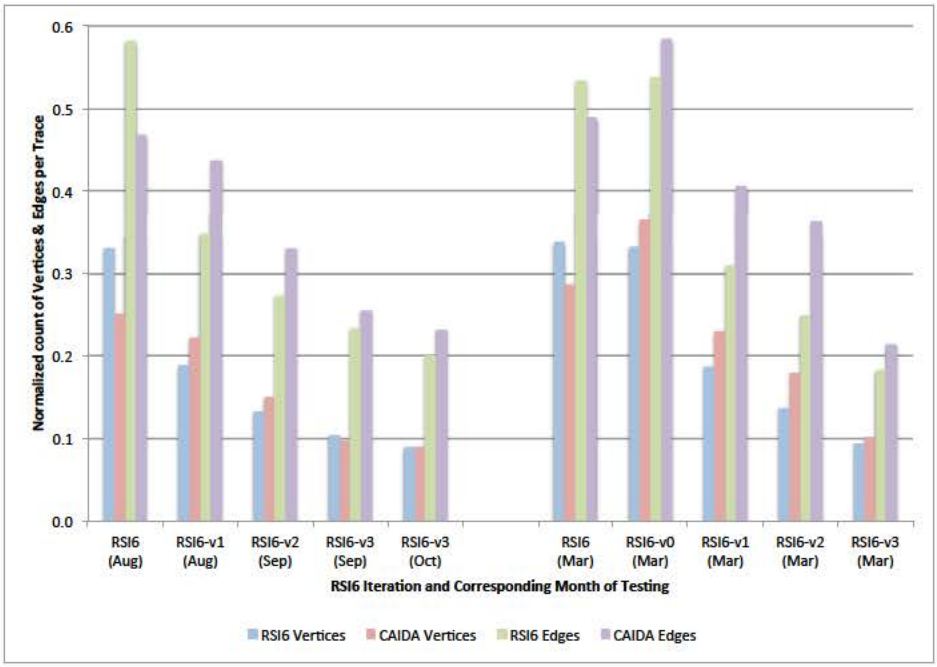


Figure 4.2: Normalized Ark and RSI6 vertex and edge data comparison

The x-axis represents the iteration of RSI6 being compared, while the y-axis depicts the normalized count of vertices and edges for both RSI6 and Ark data. It is important to note that, in order to compare similar data sets, the number of traces performed by each method was held constant. Ark data is available on a per-monitor basis and each monitor contains roughly the same number of traces. Therefore, multiple monitor files, picked at random, were selected until the total number of traces conducted was within 15% of the number of RSI6 traces listed in Table 4.2. Relative percentage differences are given in Table 4.4. The vertex and edge data was then normalized according to the actual number of traces conducted by each method and subsequently graphed. Relative comparison across all iterations show that, as revisions occurred, RSI6 was increasingly able to discover more of the topology as compared with Ark. One notable exception is the base version of RSI6, which appears to discover more topology than Ark. One possible explanation for this is that the base version of RSI6 splits prefixes using a one bit mask increment (i.e., splits the prefix in two) whereas RSI6-v1 and later versions split using an increment of four bits (the

nibble boundary). By splitting in two, the base version may be able to detect topology at a smaller granularity than later versions of RSI6. This would explain the large percentage of vertices and edges found compared with both Ark and later RSI6 version data.

Ark Traces	*RSI6 traces relative to Ark*	*Ark hours to complete*	*RSI6 hours to complete*	*RSI6 improvement relative to Ark*	*RSI6 Iteration*	
166,782	2.2%	220	21.5	90.2%	Base RSI6	2014
200,138	5.7%	264	28.5	89.2%	RSI6-v1	
325,108	2.5%	440	45.5	89.7%	RSI6-v2	
531,897	−0.7%	704	68.0	90.3%	RSI6-v3 (1)	
591,751	2.0%	792	82.0	89.6%	RSI6-v3 (2)	
162,928	−13.7%	168	42.5	74.7%	Base RSI6	2015
122,195	5.2%	126	40.0	68.3%	RSI6-v0	
203,659	−0.3%	210	49.0	76.7%	RSI6-v1	
285,119	7.5%	294	35.5	87.9%	RSI6-v2	
528,281	5.6%	588	47.0	92.0%	RSI6-v3	

Table 4.4: Ark/RSI6 comparison based on total time to completion

The second measurement of RSI6 performance relative to Ark is the time required to completely probe all BGP-announced prefixes. While this is based on a comparison between concurrent time (multiple Ark monitors operating at the same time) and serial time one RSI6 algorithm), it is important to note that every iteration of RSI6 performs this task *significantly* quicker than Ark while returning relatively similar results. Additional refinement and application of other techniques to RSI6 will most certainly provide continued improvement in its ability to discover topology.

4.2 Ground Truth Methods

Results from the direct application of RSI in IPv6 prompted the question of whether the implementation of RSI6 is flawed or if little topology exists within the prefixes RSI6 probed. To answer this question, an understanding of the underlying structure of IPv6 network prefixes is necessary. This section will discuss results of the exhaustive probing methods described in Section 3.3.

4.2.1 Exhaustive Probing with ToD

Using the methodology described in Section 3.3.1, the top twenty /32 prefixes were selected via the inference program in Table 3.3 and probed. The resulting data, namely unique and cumulative interface counts for individual monitors, was then graphically examined to identify any patterns to make inferences about the prefix structure. Results for one example prefix are displayed in Figure 4.3.

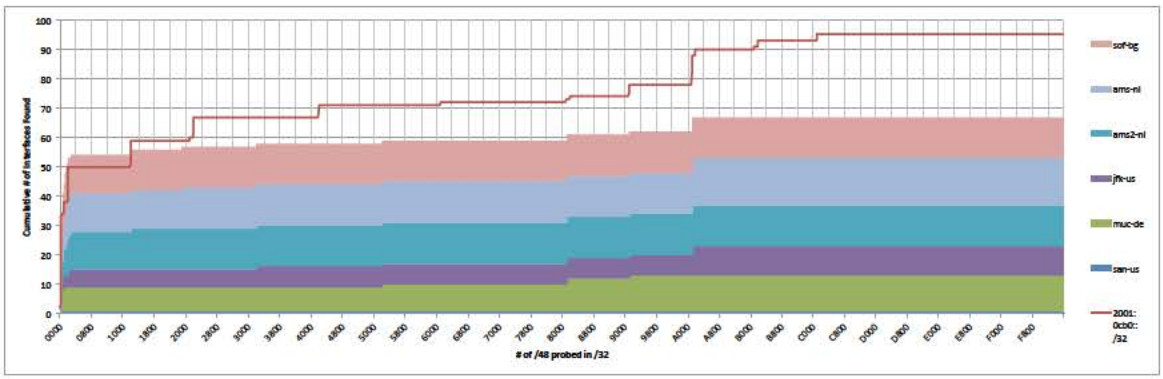


Figure 4.3: All monitor data versus select individual monitor data

The data represented in Figure 4.3 is a compilation of interface data for one specific prefix, 2001:0cb0::/32 as probed by six different monitors. This particular data set was chosen due to its large set of within-prefix interfaces and the distinct jumps in the data which indicate subnetting. As discussed in Section 3.3.1, the /32 prefix is broken down into its constituent /48s. The 16 bits that make up each unique constituent /48 are represented in Figure 4.3 along the x-axis in hexadecimal. The y-axis represents the cumulative number of unique interfaces encountered as each /48 traceroute (2^{16}, from 0x0000 to 0xFFFF) is analyzed.

The six shaded color regions in the plot represent individual data from six monitors, arranged in a stacked column. The relative height of each stack indicates the cumulative count for each monitor. For example, at roughly 0xA000, there is a large jump in the stacked graph, specifically in the height of the purple shaded region (corresponding to monitor jfk-us). This jump indicates that the jfk-us monitor encountered a large number of unique interfaces around that /48 area in the prefix. Notice that the relative height for the other shaded regions remains roughly the same as before the jump, indicating no new interfaces were seen by the other monitors. Differing numbers of interfaces between monitors highlights an advantage and a disadvantage of using a distributed system of monitors – not

every monitor will return the same topology information about a prefix. This is typically attributed to the fact that prefixes have multiple ingress points. Each monitor's probes traverse a different path, and thus a different ingress point, to reach its destination, resulting in different interfaces discovered.

The red line graphed in Figure 4.3 represents the cumulative count of interfaces across all monitors simultaneously. The probing for this prefix is distributed across all monitors, with monitors being randomly assigned a /48 to probe. The resulting interface data was then sorted according to increasing /48 number (from 0x0000 to 0xFFFF) and the cumulative count graphed. Where an individual monitor sees additional interfaces, a corresponding rise in the line graph is seen. Of note, the cumulative interface count across all monitors differs slightly from the subset of individual monitors shown due to the exclusion of data from the other 32 monitors (38 used, 6 selected for graphing). The dichotomy is that the line graph can display more complete data while the stacked column graphs more readily highlight jumps in interface count. Overall, the combined data from all monitors provides the best graphical method for inferring subnet structure. Specific to this example, the graph depicts about 16 subnets within this /32 prefix.

The key limitation with this method of graphical analysis is the large amount of data to manipulate for a single prefix. Using several individual monitor data streams (each with 65,535 lines of interface data) plus the aggregated data stream becomes extremely intensive. Further, to gain any substantive insights into ground truth subnet practices, we would like to examine the subnetting structure of *all* routed /32 IPv6 prefixes. Section 4.2.2 provides the results of large-scale testing.

4.2.2 Ark Probing

Given the roughly 6,000 /32 IPv6 prefixes on the Internet, and the intensive data probing and analysis involved in processing just one /32, it is not feasible to utilize the ToD interface to perform large-scale testing over all /32s. Using the methodology described in Section 3.3.2, large-scale probing of all /48s contained in BGP-announced /32s was conducted directly from the IPv6 Ark monitors. The resulting exhaustive probing data, comprised of just over 393 million probes, is then used to create graphical representations similar to those in Section 4.2.1.

In addition to avoiding the overhead of the ToD interface, the key difference between using ToD probing and Ark monitor probing is economies of scale – data can be analyzed for a single prefix or for all /32s. Using single prefix data, some general conclusions about typical patterns and common allocations of subnets inside a /32 can be made. Data generated from all /32s can provide a general understanding of how many subnets are typically found in a /32. Together they provide insight into subnetting practices common on the Internet, encouraging future development of an efficient method to discover topology.

Single Prefix Processing

To visualize patterns in a single prefix, two types of plots are used – a scatterplot and a line plot. The scatterplot is created using single data points to represent unique interfaces discovered by Ark probes to every /48 in a given /32. The line plot graphs the cumulative count of interfaces seen from 0x0000 to 0xFFFF (16 bits corresponding to bits 32 through 48 in the hexadecimal prefix). Figure 4.4 depicts a scatterplot from probes conducted to all $64k$ /48s within the 2001:0218::/32 prefix. This particular set of data was chosen for its large number of interfaces seen, observable patterns near nibble boundaries, and large number of likely subnets.

Similar to Figure 4.3, the x-axis represents the unique four-digit hexadecimal representation of each /48 address in 2001:0218::/32. In the case of the Figure 4.4 scatterplot, the displayed data is *all* unique interface data, not a cumulative count as in Figure 4.3. The y-axis represents the interface identifier, where the identifier x corresponds to the x'th unique interface address observed via *traceroute* for the corresponding /48 address. When two data points share the same y-value but different x-values, this indicates that the same interface was seen along two different *traceroute* destinations to different /48's within the same /32.

When probes to a /32 prefix is plotted in this manner, three particular patterns of note begin to emerge and are used to make inferences about the structure of the prefix. The first is any vertical column of localized dot groupings, indicating the possibility of subnetting in that area. The next two patterns are natural extensions of the first. The second pattern is characterized by where the vertical column falls on the x-axis compared with nibble boundaries. Referring back to Section 3.2.4, NANOG published a best common operating practice on this observed phenomenon. In short, for ease of human readability, network administrators will subnet on a nibble boundary. Figure 4.4 depicts several vertical columns

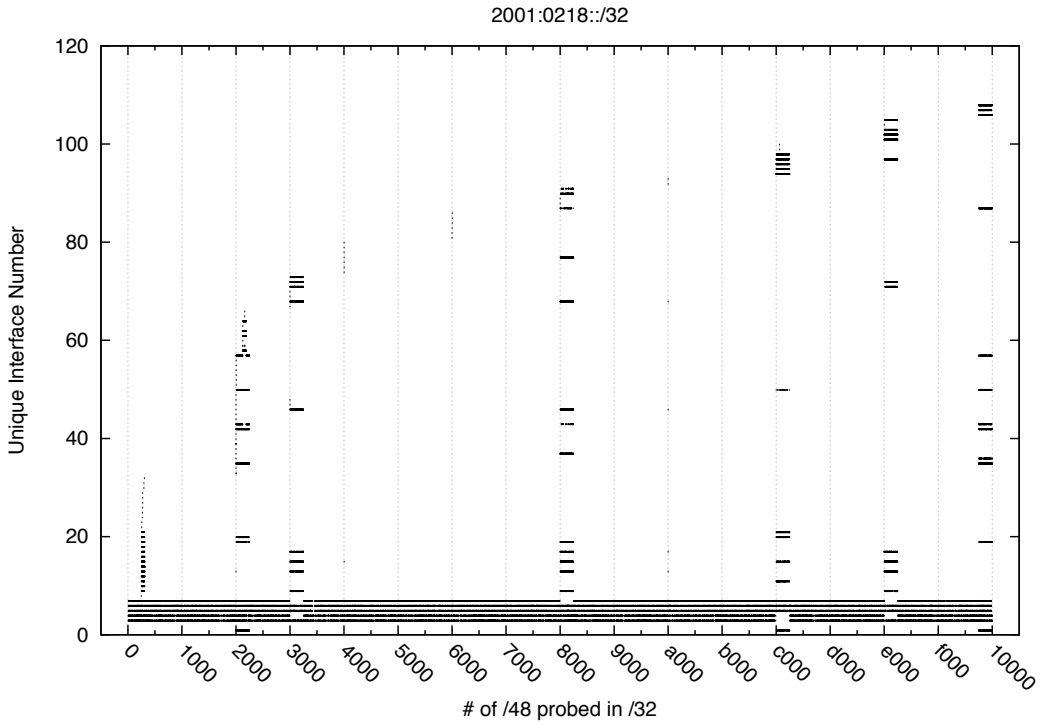


Figure 4.4: Scatterplot of interfaces in 2001:0218::/32

which have starting points that coincide with a major x-axis tick mark. This indicates the possible existence of subnetting that begins on a major nibble boundary.

The third pattern to note is characterized by the relative width of the vertical columns, corresponding to the relative size of the subnet. Wider columns tend to indicate subnetting with larger granularity (i.e., toward the /32 mask) whereas a small width would indicate smaller granularity subnetting (i.e., toward the /48 or longer mask). Finally, graphs with nearly solid horizontal lines toward the bottom indicate likely AS ingress interfaces, explaining why almost every probe into a prefix returns them. In Figure 4.4, for example, there appear to be five ingress interfaces to the AS, or at the very least, five interfaces common to all routes in the prefix. Using this type of graphical analysis, one can begin to make intelligent, albeit still limited, inferences about a prefix.

Using the same technique, Figure 4.5 depicts the prefix profile for 2a02:0d28::/32, which is

quite different from the profile shown in Figure 4.4. In this prefix, the graph is dominated by horizontal lines and very few vertical columns of dots, indicating relatively few subnets. For the very few columns that *do* exist, the narrow width suggest small subnets, most likely a few bits short of a /48 mask (e.g., /46 or /47).

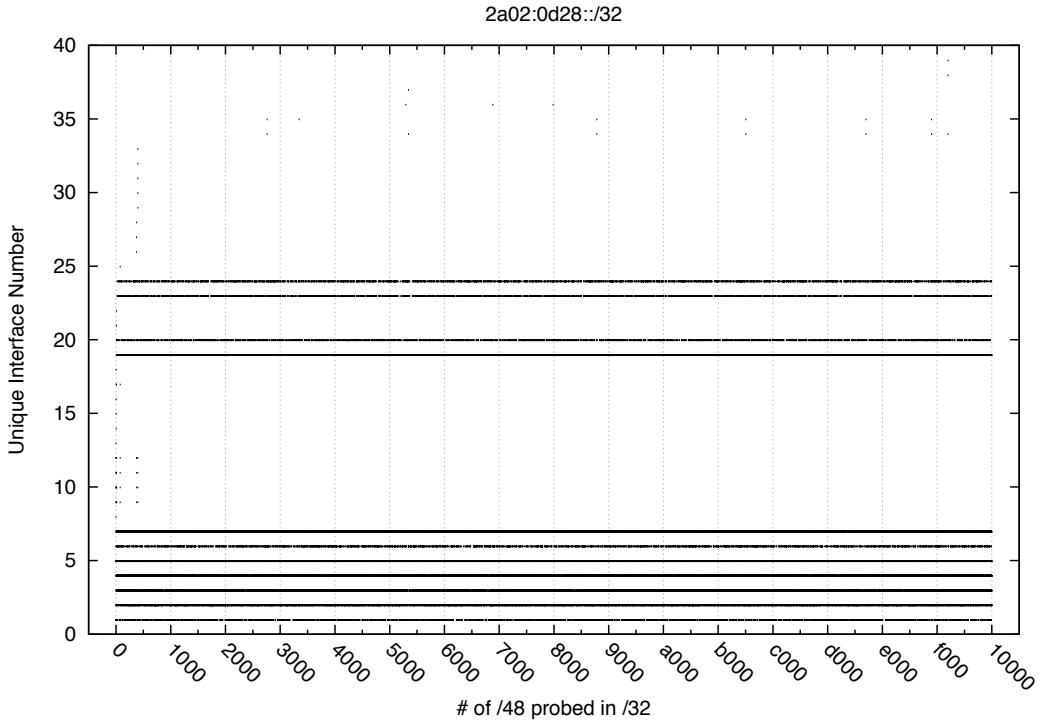


Figure 4.5: Scatterplot of interfaces in 2a02:0d28::/32

Figure 4.6 provides a different visualization of the same data as Figure 4.4. Here, the cumulative number of unique interfaces is summed from 0x0000 to 0xFFFF in hexadecimal and results are plotted as a line graph. Similar to previous figures, the x-axis represents the unique /48 number while the y-axis represents the cumulative count of interfaces seen. This visualization is important because, in some cases, unique interfaces are not as evident on a scatterplot as they are with a cumulative count plot. The nuanced increases in interface numbers can be easily seen on a line graph. Relative jumps in cumulative interfaces can suggest the entrance of probes into a new subnet, as shown in Figure 4.6. Applying this inference, counting jumps on the graph is akin to counting subnets. In this case, Figure 4.6

depicts roughly 19 subnets where the actual number inferred by the ground truth method presented in Table 3.4 is 38. The difference is attributed to minuscule jumps that may not be easily visible when looking at the graph, but large enough that our ground truth method perceives them as possible subnets.

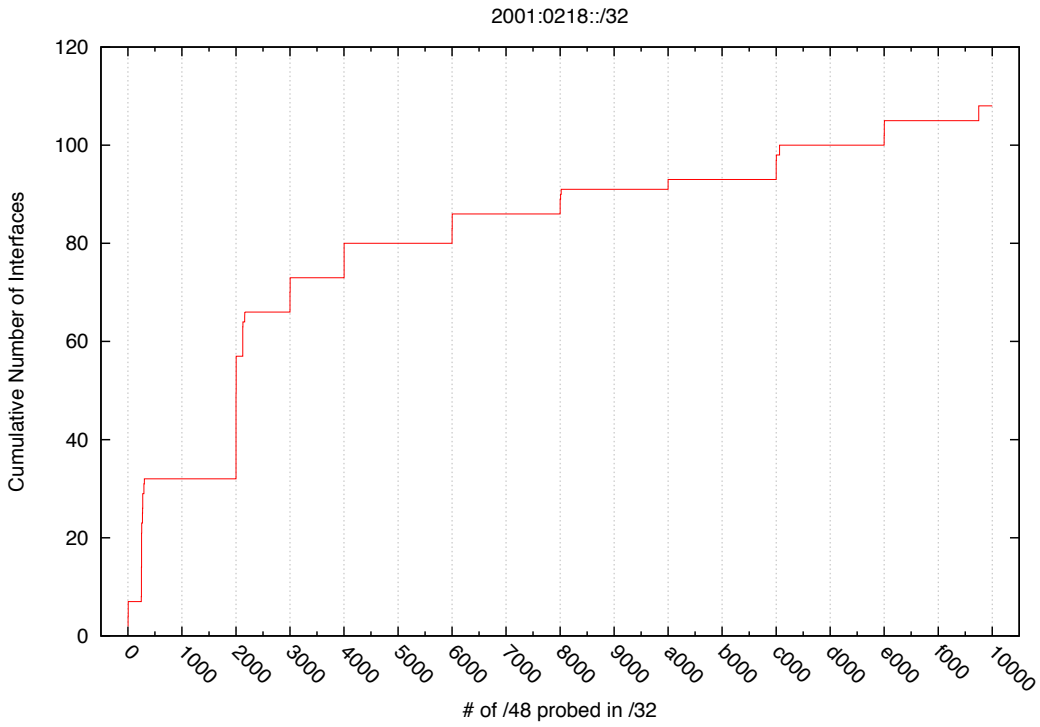


Figure 4.6: Cumulative interfaces in 2001:0218::/32

Figure 4.7 depicts how the scatterplot of one prefix, 2001:0728::/32, transforms to the cumulative interface graph. As unique interfaces are discovered, they are assigned a number corresponding to values on the y-axis. Large jumps in unique interfaces seen on the left correspond to a large jump in the line graph on the right. Using this "jump," we infer that probing has just encountered a new subnet. Given that several jumps occur near major nibble boundaries, it is reasonable to assume that a subnet exists at that location.

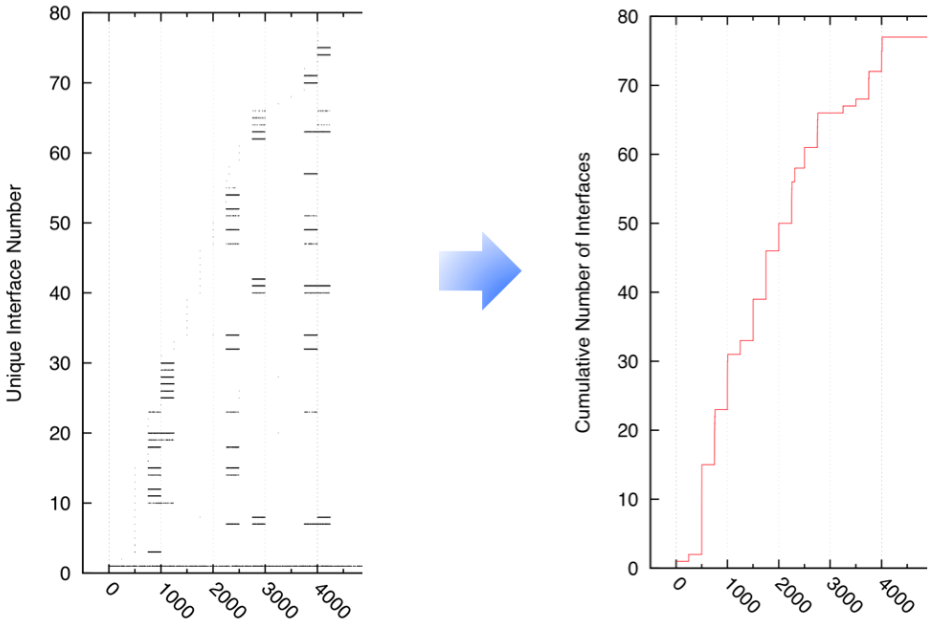


Figure 4.7: Transformation of the unique interface scatterplot to cumulative interfaces for 2001:0728::/32

All /32 Prefix Processing

Another key metric that lends insight to how IPv6 prefixes are structured is the relative number of subnets each prefix contains. Combined with observed patterns in the graphic representation of single prefix data, general insights about the structure of all /32s can be made. Used effectively, these insights can be transformed into future efficient topology probing techniques.

Using the methodology of Section 3.3.2 and the ground truth analysis method in Table 3.4, the gathered data from Ark's exhaustive probing is analyzed. Specifically, data regarding inferred subnets is extracted and used to construct a cumulative distribution function graph, shown in Figure 4.8. The graph depicts the number of subnets contained within a prefix on the x-axis (using a base-2 logarithmic scale) with the cumulative percentage of all analyzed subnets on the y-axis. According to the graph, the ground truth method discussed in Table 3.4 infers one or fewer subnets in approximately 65% of prefixes, while the top 5% have 16 or more subnets. Taking the analysis one step further, approximately 90% of all /32 prefixes contain eight or fewer subnets between /32 and /48.

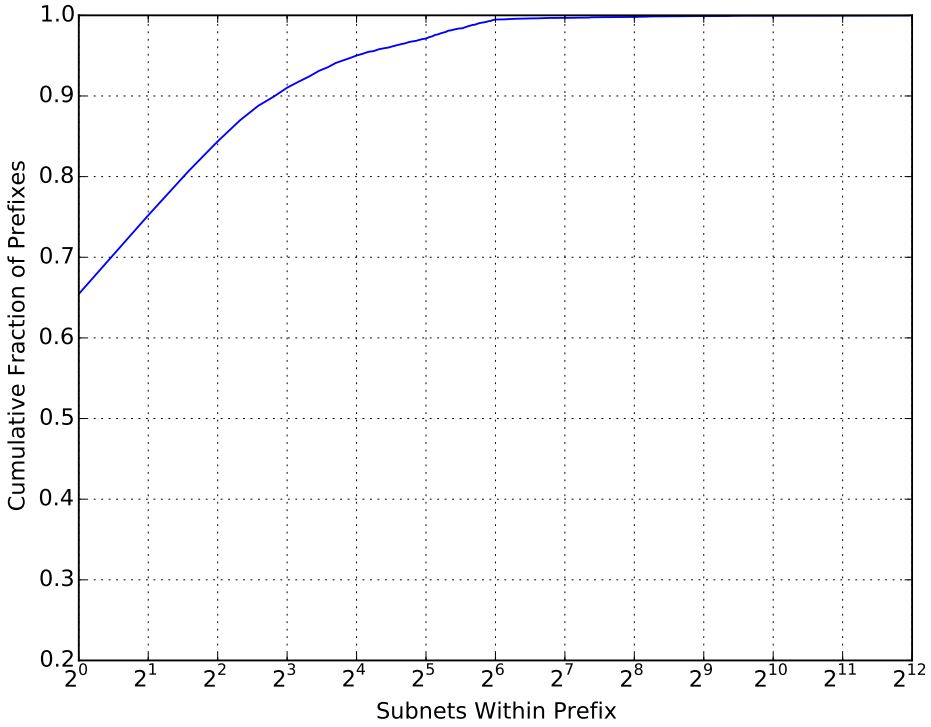


Figure 4.8: Cumulative distribution of subnets in all /32 prefixes

Of note to this research is that, given a random /32 prefix, probing using an intelligent algorithm like RSI6 may terminate prematurely due to the lack of sufficient subnet structure. Based on the data in Figure 4.8, only 10% of all prefixes provide sufficient structure suitable for probing with an algorithm that uses a strategy similar to RSI6. In general, the low number of subnets across the majority of /32s may be attributed to the relative infancy of IPv6. As IPv6 adoption continues to increase, future IPv6 subnetting may provide sufficient structure for intelligent probing. In any case, the research conducted here will provide a solid base with which to further develop intelligent topology mapping algorithms.

4.3 Validation of Select Prefixes

To further support the ground truth inference methods presented in Section 4.2.2, we sought ground truth for 14 prefixes in three ASs by querying the servicing ISPs. Unfortunately, we did not receive any responses to our queries.

CHAPTER 5:
Conclusions

This research evaluates and implements an efficient IPv4 topology mapping algorithm in IPv6. In modifying and testing different techniques used by RSI6, this research shows that, even though RSI4 was efficient in discovering topology, the same methodology does not produce equally efficient results in IPv6. Furthermore, efforts to uncover a subset of ground truth topology via exhaustive probing of /48s within BGP-announced /32 prefixes discovered several key patterns for future validation with ISPs. These results provide an important basis for future work into efficient IPv6 topology mapping techniques.

5.1 Limitations

While results from research on IPv4 techniques gives us insight into some of the challenges that can be encountered, IPv6 introduces new challenges. This section describes some of the limitations encountered during the development and implementation of both the RSI6 and ground truth methods.

5.1.1 IPv6 Infancy

As discussed in Chapter 2, IPv6, while existing for over two decades, is still relatively new in terms of deployment and adoption as compared to IPv4. With the recent IPv4 address exhaustion and a declining ability to extend the life of the IPv4 address space further, IPv6 deployment has enjoyed exponential growth for the last several years. Even with this rapid development, the expansive IP space associated with IPv6 makes it difficult to map the topology due to large pockets of unused and unallocated blocks of IP addresses. Specifically, the terminating conditions RSI6 uses are predicated upon not discovering new interfaces, a condition that frequently occurs. With IPv6, the probability that RSI6 will terminate prematurely for a given prefix, thereby missing topology within that prefix, is high. However, we might expect better performance from RSI6 in the future as IPv6 adoption continues and more of the IPv6 address space is utilized.

5.1.2 Vantage Point (VP) Selection

While a distributed network of monitors (or VPs) allows multiple points from which to probe, we find that different VPs experience different reachability to a single destination. In some cases, probes to a certain IPv6 address from one VP are able to traverse into the target prefix whereas probes from another geographically separate VP are not. This implies a dependence on the VP when obtaining topology. While per-VP differences are to be expected, a VP that is unable to reach the destination network can be problematic. Specifically, the concern is the return of no results when, in fact, the probe should return results from the target prefix. In the case of RSI6, probing of the parent prefix would terminate prematurely and report that it discovered no further topology, which is incorrect. The diversity afforded by different VPs has also been observed in IPv4 and a technique was developed to best select VPs for a given target [25]. Future work should investigate using intelligent VP selection techniques in IPv6.

5.1.3 Load Balancing

Redundancy and consistency are especially important properties of the operation of the Internet. With the rapid expansion of global Internet users comes increased congestion. Network administrators commonly enable load-balancing between sets of routers in order to increase overall capacity and to share the load under heavy-use conditions. Flow tuple hashing algorithms determine which path traffic traverses, which can affect the path that our probe traffic traverses. The consistency of results depends on the interfaces that probes see, hence load balancing impacts topology mapping. If RSI6 detects an interface that it had not encountered before, it assumes subnetting and splits the prefix. In reality, the new interface discovered may be part of a load-balanced path rather than an indication of subnetting. In this case, RSI6 would detect additional topology where none exists. While Paris-*traceroute* avoids such false inferences in IPv4 [36], a better understanding of the extent and impact of load balancing in IPv6 would be a valuable future contribution.

5.1.4 Inference Validation

Attempting to determine ground truth about a network prefix using standard network tools such as *ping* and *traceroute* is an impossible task. These are inference tools and are subject to distortions due to their attachment point and the remote network policy. The largest hurdle is to determine an accurate method of inferring a network's structure based on the

limited information returned. Once a metric is determined and implemented, there is still no guarantee that the inference is accurate.

However, only through validation of results can a researcher be confident in their inferences and methods. Unfortunately, obtaining ground truth or validation from the large organizations that run the Internet can be difficult. Not only are there multiple tier-level ISPs, but much of the information regarding structure of their systems is proprietary and not releasable. This limits a researcher to two options: using a small sample of validated results to make large-scale conclusions or making comparisons between many prefixes to characterize the relative size of a particular prefix. Without concrete data, a ground truth inference is limited to relative comparisons between methods to see which performs better, however, it is not the same as knowing which is correct or better at inferring topology.

5.1.5 Exhaustive Probing

Given the scope of this research, only prefix lengths between /32 and /48 were exhaustively probed in order to form a better baseline about deployed IPv6 network structure. Exhaustive probing of all /48s contained in every BGP-announced /32 (methodology in Section 3.3.2) took roughly three months to complete using Ark. The decision to examine prefixes between /32 and /48 was arbitrary, but does not preclude the possibility that subnetting occurs at prefix lengths greater than /48. In fact, the IPv6 space between /48 and /64 contains just as many possibilities as what was presented here. Additionally, subnetting indicated by the results of exhaustive probing could occur in either case with no evidence of where the subnetting actually exists (i.e., between /32 and /48, between /48 and /64, or further into the prefix). Just as significant, announced prefixes shorter than /32 were excluded from testing and left for future work.

5.2 Future Work

The techniques and implementation presented in this research provide a necessary starting point for the development of IPv6 topology probing techniques. This section describes some of the other necessary areas of development to further the design of an efficient IPv6 topology mapping algorithm.

5.2.1 Other IPv4 Techniques

RSI6 is an adaptation of only one of the proven efficient techniques developed for IPv4. There are several advanced probing techniques that require research and implementation into IPv6. Specifically, VP selection, IPS, and Sequence Completion applied to IPv6 are possible areas of future research.

VP Selection

As described in Section 5.1.2, VP selection becomes a factor when attempting to characterize the performance of efficient algorithms. Specifically, it affects the availability of results of probing, which can compromise the inference of topological data. Prior work developed VPS (Section 2.2.1) to intelligently spread the probing of a prefix across multiple VPs, ensuring the maximum amount of topology information is discovered from the aggregate of all probes sent to a given prefix. In this way, IPv4 VPS is able to maximize the availability of results from probes and use this to correctly discover topological information. VPS should be adapted to operate in IPv6 and work in conjunction with RSI6 to maximize the resulting information and discovered topology.

Multiple Ingress Points

The concept of using multiple VPs to probe a prefix is based on the assumption that a prefix most likely has multiple ingress routes. Each ingress provides a different path to the probe destination, similar to the effect seen with VPS. In IPv4, a technique called IPS takes this one step further and intelligently selects VPs based on past success and inter-VP path diversity in accessing a prefix. As shown in Section 2.2.1, this technique, coupled with RSI, provided an effective combination in maximizing the topology discovered. Conversion to IPv6, along with an IPv6 version of VPS, would provide researchers a solid base to develop an efficient IPv6 mapping algorithm.

Sequence Completion

In the absence of guidance, the human factor of network administration will inevitably lead to predictable patterns in addressing schemes. In IPv4, as discussed in Section 2.2.2, Raytheon and BBN Technologies performed research into exploiting IPv4 sequence completion. Results in IPv4 suggest application of this concept is prevalent in networks to

some degree. Given the similar operational nature of IPv4 and IPv6, future work needs to be performed to determine whether sequence completion works today.

5.2.2 Heuristic Methods

The human factor not only plays into selection of subnet structure, but recent research in [37] heuristically explores the most common patterns in the lower 64 bits of IPv6 router interface addresses. Using this data, an algorithm could detect these common addresses and dynamically tailor probes to more efficiently detect topology. While the work in [37] is promising, it remains a preliminary step forward in the development of efficient techniques in IPv6. Future work to combine RSI6 and heuristics may prove to be an effective solution to the IPv6 topology mapping challenge.

5.2.3 Subnet Boundaries

As discussed in Section 3.2.4, published best practices recommend network administrators subnet on a nibble boundary, mainly for human readability. As with sequence completion, human interaction begets predicable patterns which could be utilized. The research conducted here only slightly touches on this subject in an attempt to further refine RSI6. Additional research into the size of commonly deployed IPv6 subnet boundaries could provide key insights that might lead to an algorithm that can exploit these human-based patterns more efficiently.

5.2.4 Ground Truth in Different Prefix Lengths

As discussed in Sections 3.3 and 5.1.5, the research presented here focused on the $64k$ constituent /48 sub-networks of each BGP-announced /32 prefix. The resulting 393 million probes took approximately three months to complete. Given that subnetting can occur anywhere in a prefix due to CIDR-notation, it is logical that at each level of distribution, IP address blocks are being subnetted. This assumption leads to the conclusion that, not only is there subnetting in the data we collected here, but additional subnetting occurs in prefix lengths greater than /48 as well. Additional research, coupled with the data gathered here, could provide a more complete ground truth picture that can be used for future validation of efficient topology mapping algorithms.

5.2.5 Using Results of Ground Truth

Our research into ground truth methods was limited in scope and validation of results. Given future research and additional validation of full-prefix ground truth (ours was only partial-prefix ground truth), conclusions can be made about how the IPv6 address space is being partitioned in practice. Using this information, techniques can be developed retroactively to detect the patterns observed through ground truth. In effect, this is attacking the challenge of topology discovery backwards by developing an algorithm based on a subset of known topology in order to better map unknown topology. The benefit is that future topology discovered by the newly created algorithm will indeed be efficient and accurate.

5.3 Concluding Remarks

With the imminent exhaustion of the IPv4 address space, adoption of IPv6 is occurring at a rapid pace. As systems migrate to IPv6, IPv6 will become an increasingly important component of the cyber domain. Hence, IPv6 is becoming a priority for the DOD and other government agencies. Work on discovering and understanding IPv6 networks and topology will therefore become increasingly important. Given the vastness of the IPv6 address space, continued research into efficient mapping and characterization algorithms and techniques is a necessity to ensure continued dominance in both offensive and defensive capabilities.

References

[1] History of the Internet. (n.d.) *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/History_of_the_Internet

[2] ARIN IPv4 countdown plan. (n.d.). American Registry for Internet Numbers. [Online]. Available: https://www.arin.net/resources/request/ipv4_countdown.html

[3] State of the Union 2015. (2015, Jan. 20). [Online]. Available: http://www.cnn.com/2015/01/20/politics/state-of-the-union-2015-transcript-full-text/

[4] B. Obama, *The National Security Strategy of the United States of America*. Washington DC: The White House, May 2010.

[5] IPv6 Adoption. (n.d.). Google Statistics. [Online]. Available: https://www.google.com/intl/en/ipv6/statistics.html

[6] IPv6 Enabled Networks. (n.d.). Ripe Network Coordination Centre. [Online]. Available: http://v6asns.ripe.net/v/6?s=_ALL;s=CH;s=DE;s=JP;s=FR;s=US;s=BE;s=PE

[7] R. Beverly, A. Berger, and G. Xie, "Primitives for Active Internet Topology Mapping: Toward High-Frequency Characterization," in *10th ACM SIGCOMM Conference on Internet Measurement*, 2010, pp. 165–171.

[8] IPv6 Topology Dataset. (2014, Jun). Center for Applied Internet Data Analysis. [Online]. Available: http://www.caida.org/data/active/ipv6_allpref_topology_dataset.xml

[9] G. Baltra, "Efficient Strategies for Active Interface-Level Network Topology Discovery," M.S. thesis, Dept. Comp. Sci., Naval Postgraduate School, Monterey, CA, 2013.

[10] J. Postel, "Internet Protocol," RFC 791 (INTERNET STANDARD), Internet Engineering Task Force, Sep. 1981, updated by RFCs 1349, 2474, 6864. [Online]. Available: http://www.ietf.org/rfc/rfc791.txt

[11] Total Population. (n.d.). World Bank. [Online]. Available: http://data.worldbank.org/indicator/SP.POP.TOTL

[12] How many things are currently connected to the "Internet of Things" (IoT). (2013, Jan. 7). *Forbes*. [Online]. Available: http://www.forbes.com/sites/quora/2013/01/07/how-many-things-are-currently-connected-to-the-internet-of-things-iot/

[13] G. Huston, “IPv4 address report,” Mar 2015. [Online]. Available: http://www.potaroo.net/tools/ipv4/

[14] J. Postel, “Address mappings,” RFC 796 (Historic), Internet Engineering Task Force, Sep. 1981. [Online]. Available: http://www.ietf.org/rfc/rfc796.txt

[15] V. Fuller and T. Li, “Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan,” RFC 4632 (Best Current Practice), Internet Engineering Task Force, Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4632.txt

[16] K. Egevang and P. Francis, “RFC 1631 The IP Network Address Translator (NAT),” Internet Engineering Task Force, May 1994. [Online]. Available: http://tools.ietf.org/html/rfc1631

[17] American Registry for Internet Numbers. (2014, May). IPv4 depletion and IPv6 adoption today. [Online]. Available: https://www.arin.net/knowledge/v4_deplete_v6_adopt.pdf

[18] OECD, “The Internet in Transition: The State of the Transition to IPv6 in Today’s Internet and Measures to Support the Continued Use of IPv4,” *OECD Digital Economy Papers*, no. 234, 2014. [Online]. Available: http://dx.doi.org/10.1787/5jz5sq5d7cq2-en

[19] R. Hinden and S. Deering, “IP Version 6 addressing architecture,” RFC 4291 (Draft Standard), Internet Engineering Task Force, Feb. 2006, updated by RFCs 5952, 6052, 7136, 7346, 7371. [Online]. Available: http://www.ietf.org/rfc/rfc4291.txt

[20] R. Hinden and S. Deering, “IP Version 6 addressing architecture,” RFC 1884 (Historic), Internet Engineering Task Force, Dec. 1995, obsoleted by RFC 2373. [Online]. Available: http://www.ietf.org/rfc/rfc1884.txt

[21] J. Czyz, M. Allman, J. Zhang, S. Iekel-Johnson, E. Osterweil, and M. Bailey, “Measuring IPv6 adoption,” in *ACM SIGCOMM*, Aug. 2014.

[22] A. Dhamdhere, M. Luckie, B. Huffaker, A. Elmokashfi, E. Aben *et al.*, “Measuring the deployment of IPv6: Topology, routing and performance,” in *Proceedings of the 2012 ACM Cnference on Internet Measurement*. ACM, 2012, pp. 537–550.

[23] N. Spring, R. Mahajan, and D. Wetherall, “Measuring ISP topologies with Rocketfuel,” in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM ’02. New York, NY, USA: ACM, 2002, pp. 133–145. [Online]. Available: http://doi.acm.org/10.1145/633025.633039

[24] k. claffy, Y. Hyun, K. Keys, M. Fomenkov, and D. Krioukov, "Internet mapping: from art to science," in *IEEE DHS Cybersecurity Applications and Technologies Conference for Homeland Security (CATCH)*, Watham, MA, Mar 2009, pp. 205–211.

[25] G. Baltra, R. Beverly, and G. G. Xie, "Ingress point spreading: A new primitive for adaptive active network mapping," in *Proceedings of the Fifteenth Passive and Active Measurement Conference*, March 2014.

[26] B. Donnet, P. Raoult, T. Friedman, and M. Crovella, "Efficient algorithms for large-scale topology discovery," in *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '05. New York, NY, USA: ACM, 2005, pp. 327–338. [Online]. Available: http://doi.acm.org/10.1145/1064212.1064256

[27] B. Eriksson, G. Dasarathy, P. Barford, and R. Nowak, "Efficient network tomography for internet topology discovery," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 931–943, Jun. 2012. [Online]. Available: http://dx.doi.org/10.1109/TNET.2011.2175747

[28] X. Lang, G. Zhou, C. Gong, and W. Han, "Dolphin: The measurement system for the next generation Internet," in *Communications, Internet, and Information Technology*, October 2005. [Online]. Available: http: //ipv6backbone.blogspot.com/2008/10/dolphin-measurement-system-for-next.html

[29] Lumeta Corporation. (2014, June). Lumeta IPsonar: Data Sheet. [Online]. Available: http://www.lumeta.com/pdfs/Lumeta_IPsonar_datasheet.pdf

[30] R. Barnes, R. Altmann, and D. Kerr, "Mapping the great void: Smarter scanning for IPv6," in *CAIDA Workshop on Active Internet Measurements*. San Diego, CA: BBN Technologies, 2012.

[31] Center for Applied Internet Data Analysis. (2014, Nov. 4). Ark measurement infrastructure. [Online]. Available: http://www.caida.org/projects/ark/

[32] Center for Applied Internet Data Analysis. (2015, Mar. 10). CAIDA monitors. [Online]. Available: http://www.caida.org/data/monitors/monitor-map-ark.xml

[33] Trustees of Princeton University. (2015, Mar. 5). Planetlab design notes. [Online]. Available: https://www.planet-lab.org/doc/pdn

[34] Ripe Network Coordination Centre. (2015, Mar. 5). Ripe Atlas: REST API. [Online]. Available: https://atlas.ripe.net/docs/rest/

[35] C. Grundemann, A. Hughes, and O. DeLong. (2012, Feb. 6). Best current operational practices - IPv6 subnetting (v1). [Online]. Available: http://bcop.nanog.org/index.php/IPv6_Subnetting

[36] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira, "Avoiding traceroute anomalies with Paris traceroute," in *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*. ACM, 2006, pp. 153–158.

[37] M. Gray, "Discovery of IPv6 Router Interface Addresses via Heuristic Methods," unpublished.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California Washington, DC